UNIVERSIDADE FEDERAL DO MARANHÃO
Programa de Pós-Graduação em Engenharia Elétrica

Juan Daniel Ferreira Ibanez

UM *FRAMEWORK* BASEADO EM *DATA LAKEHOUSE* PARA ARMAZENAMENTO, ANÁLISE E INTEGRAÇÃO DE DADOS EM SISTEMAS ENERGÉTICOS RENOVÁVEIS

São Luís - MA
2026

Juan Daniel Ferreira Ibanez

UM *FRAMEWORK* BASEADO EM *DATA LAKEHOUSE* PARA ARMAZENAMENTO, ANÁLISE E INTEGRAÇÃO DE DADOS EM SISTEMAS ENERGÉTICOS RENOVÁVEIS

Dissertação apresentada como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, ao Programa de Pós-Graduação em Engenharia Elétrica, da Universidade Federal do Maranhão.

Programa de Pós-graduação em Engenharia Elétrica

Universidade Federal do Maranhão

Orientador: Prof. Dr. Denivaldo Cicero Pavão Lopes

São Luís - MA

2026

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Diretoria Integrada de Bibliotecas/UFMA

Ibanez, Juan Daniel Ferreira.
UM FRAMEWORK BASEADO EM DATA LAKEHOUSE PARA ARMAZENAMENTO, ANÁLISE E INTEGRAÇÃO DE DADOS EM SISTEMAS ENERGÉTICOS RENOVÁVEIS / Juan Daniel Ferreira Ibanez. - 2026.
151 p.

Orientador(a): Denivaldo Cicero Pavão Lopes.
Dissertação (Mestrado) - Programa de Pós-graduação em Engenharia Elétrica/ccet, Universidade Federal do Maranhão, São Luís, 2026.

1. Data Lakehouse. 2. Fontes Renováveis. 3. Séries Temporais. 4. Perfiladores. 5. Transição Energética. I. Lopes, Denivaldo Cicero Pavão. II. Título.

Juan Daniel Ferreira Ibanez

UM *FRAMEWORK* BASEADO EM *DATA LAKEHOUSE* PARA ARMAZENAMENTO, ANÁLISE E INTEGRAÇÃO DE DADOS EM SISTEMAS ENERGÉTICOS RENOVÁVEIS

Dissertação apresentada como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica, ao Programa de Pós-Graduação em Engenharia Elétrica, da Universidade Federal do Maranhão.

Trabalho Enviado para correção. São Luís - MA, 16 de Dezembro de 2025.

Prof. Dr. Denivaldo Cicero Pavão Lopes, UFMA.
Orientador

Prof. Dr. Francisco Jóse Silva e Silva, UFMA.
Examinador Interno

Prof. Dr. Ricardo de Andrade Lira Rabelo, UFPI.
Examinador Externo

Prof. Dr. Felipe Mendonça Pimenta, UFSC.
Examinador Externo

São Luís - MA
2026

Dedicatória

Para minha mãe, Lucienny de S. Ferreira.

Para minha família e amigos.

Obrigado por tudo.

Agradecimentos

Agradeço a Deus pela dádiva da vida, pela saúde e discernimento ao longo desta jornada, pela perseverança nos dias difíceis e pelas portas que se abriram quando tudo parecia incerto.

Aos meus pais, Lucienny de S. Ferreira e Roler C. V. Ibanez R., pelo apoio constante, mesmo à distância, pelas mensagens de incentivo, pela confiança silenciosa que me fez seguir adiante e por me lembrarem, sempre, de onde venho e por que caminho.

À amiga e companheira, Raiane Rodrigues, pela parceria paciente e leal: pela escuta nas madrugadas, pelas conversas que recolocaram as ideias no lugar, pela alegria nos intervalos e pela calma que me ajudou a continuar quando o cansaço pesou.

Aos amigos e colegas que conheci no Instituto de Energia Elétrica, Aline Costa, Davi Borges, Diego Cosme, Diogo Pereira, Fabricio Santos, Hellen Souza, Isabelle Oliveira, Leonilson Veras, Nerval Junior, Rafael Quezada, Rafael Veras e Wender Figueredo, pelos debates técnicos, pelos códigos compartilhados, pelas idas ao laboratório, pelos ajustes de última hora e pelo companheirismo que transforma trabalho em aprendizado.

A todos os familiares e amigos não citados diretamente, obrigado pelas orações, pelas mensagens rápidas que fizeram diferença, pelas visitas, cafés e pela confiança em cada passo desta etapa.

Aos professores do PPGEE, em especial aos professores Osvaldo Saavedra, Shigeaki Lima e José Gomes de Matos, Silvangela Barcelos e aos colegas de turma, pelas aulas exigentes, seminários, correções criteriosas e discussões que expandiram meus horizontes. Levo comigo, destes dois anos, a base do que pretendo construir adiante.

Ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Maranhão, pelo suporte institucional, pela acolhida acadêmica e pela oportunidade de desenvolver esta pesquisa com rigor e propósito.

À Fundação de Amparo à Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão (FAPEMA), pelo financiamento e pela confiança no potencial desta investigação, apoio sem o qual este trabalho não teria avançado no mesmo ritmo e qualidade.

Ao Conselho Nacional de Desenvolvimento Científico do Maranhão - CNPq, pelo financiamento do projeto "INSTITUTO NACIONAL DE CIÊNCIA E TECNOLOGIA EM ENERGIAS OCEÂNICAS E FLUVIAS", Processo Número 465672/2014-0 e Processo Número 408339/2024-1.

À Universidade Federal do Maranhão, pela infraestrutura disponibilizada, laboratórios, bibliotecas, espaços de estudo e equipamentos, que possibilitaram experimentar, testar,

errar e acertar com segurança e método.

Por fim, ao meu orientador, Denilvado Lopes, pela paciência nas idas e vindas, pelo rigor científico, pelas perguntas que importam e pelos conselhos na hora certa. Sua orientação deu direção e profundidade a este trabalho.

"O que sabemos é uma gota;
o que ignoramos é um oceano."
(Isaac Newton)

Resumo

Esta dissertação propõe o FIDLER, um *framework* baseado em Data Lakehouse para armazenamento, integração e análise de dados de energias renováveis, com foco em recursos eólicos e maremotrizes no estado do Maranhão. A pesquisa é motivada pela necessidade de soluções eficientes para lidar com a heterogeneidade, o volume e a variedade dos dados provenientes de equipamentos de medição como LIDAR, SODAR, ADCP e torres micrometeorológicas. A arquitetura é estruturada em camadas Bronze, Prata e Ouro, preservando a semântica temporal dos dados energéticos e promovendo governança, escalabilidade e integração com ferramentas analíticas modernas. O estudo incorpora extensões como TimescaleDB e PostGIS para otimização de consultas e georreferenciamento, além de definir estratégias específicas para ingestão, padronização e enriquecimento dos dados. A proposta é validada com base em campanhas reais conduzidas no Maranhão, evidenciando seu potencial para subsidiar análises de séries temporais, previsões de geração e apoio à tomada de decisão na transição energética regional.

Palavras-chave: Data Lakehouse; Fontes Renováveis; Séries Temporais; Perfiladores; Transição Energética; Armazenamento de Dados; Integração de Dados Heterogêneos e Governança de Dados.

Abstract

This dissertation proposes FIDLER, a Data Lakehouse-based framework for storing, integrating, and analyzing renewable energy data, focusing on wind and tidal resources in the state of Maranhão, Brazil. The research is driven by the need for efficient solutions to handle the heterogeneity, volume, and variety of data generated by measurement equipment such as LIDAR, SODAR, ADCP, and micrometeorological towers. The architecture is structured in Bronze, Silver, and Gold layers, preserving the temporal semantics of energy data while promoting governance, scalability, and integration with modern analytical tools. The study includes database extensions such as TimescaleDB and PostGIS to optimize queries and support georeferencing, and it defines specific strategies for ingesting, standardizing, and enriching the data. The framework is validated using real-world campaigns conducted in Maranhão, demonstrating its potential to support time-series analysis, generation forecasting, and decision-making for the regional energy transition.

Keywords: Data Lakehouse; Renewable Sources; Time-Series Data; Profilers; Energy Transition; Data Storage; Heterogeneous Data Integration and Data Governance.

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ABAC	*Attribute-Based Access Control*
ACID	*Atomicidade, Consistência, Isolamento e Durabilidade*
ACM	*Association for Computing Machinery*
ADCP	*Acoustic Doppler Current Profiler*
AES-256	*Advanced Encryption Standard (256 bits)*
AHP	*Analytic Hierarchy Process (Processo de Hierarquia Analítica)*
AIS	*Automatic Identification System*
ANSI	*American National Standards Institute*
API	*Application Programming Interface (Interface de Programação de Aplicações)*
BI	*Business Intelligence*
CBA	*Congresso Brasileiro de Automática*
CPU	*Central Processing Unit (Unidade Central de Processamento)*
CRUD	*Create, Read, Update, Delete*
CSV	*Comma-Separated Values*
Delta	*Delta Lake (formato de tabela transacional)*
ELT	*Extract, Load, Transform*
ETL	*Extract, Transform, Load*
EPE	*Empresa de Pesquisa Energética*
FAIR	*Findable, Accessible, Interoperable and Reusable*
FAPEMA	*Fundação de Amparo à Pesquisa e ao Desenvolvimento Científico e Tecnológico do Maranhão*
FIDLER	*Framework of Intelligent Data Lakehouse for Energy Resource*
GCS	*Google Cloud Storage*

GB	*Gigabyte*
GRU	*Gated Recurrent Unit (Unidade Recorrente com Portas)*
GW	*Gigawatt*
HMR	*Hot Module Replacement*
HTTP	*Hypertext Transfer Protocol*
IA	*Inteligência Artificial*
IA/ML	*Inteligência Artificial / Aprendizado de Máquina*
IAM	*Identity and Access Management*
IDE	*Integrated Development Environment (Ambiente de Desenvolvimento Integrado)*
IDS	*Intrusion Detection System (Sistema de Detecção de Intrusões)*
IEC	*International Electrotechnical Commission (Comissão Eletrotécnica Internacional)*
IEA	*International Energy Agency (Agência Internacional de Energia)*
IEEE	*Institute of Electrical and Electronics Engineers*
INEOF	*Instituto Nacional de Energias Oceânicas e Fluviais*
INMET	*Instituto Nacional de Meteorologia*
IOPS	*Input/Output Operations Per Second*
JSON	*JavaScript Object Notation*
JWT	*JSON Web Token*
LIDAR	*Light Detection and Ranging*
LSTM	*Long Short-Term Memory*
ML	*Machine Learning (Aprendizado de Máquina)*
MLOps	*Machine Learning Operations*
MVC	*Model-View-Controller (Modelo-Visão-Controlador)*
NetCDF	*Network Common Data Form*
ODS	*Objetivos de Desenvolvimento Sustentável*

OGC	*Open Geospatial Consortium*
OLAP	*Online Analytical Processing*
OLTP	*Online Transaction Processing*
OMG	*Object Management Group*
ONS	*Operador Nacional do Sistema Elétrico*
OAuth2	*Open Authorization 2.0*
ORC	*Optimized Row Columnar*
PPA	*Power Purchase Agreement*
PPGEE	*Programa de Pós-Graduação em Engenharia Elétrica*
PostGIS	*PostgreSQL Geographic Information System*
RBAC	*Role-Based Access Control*
REST	*Representational State Transfer (Transferência Representacional de Estado)*
REWS	*Rotor Equivalent Wind Speed*
RPO	*Recovery Point Objective*
RTO	*Recovery Time Objective*
S3	*Simple Storage Service*
SBSE	*Simpósio Brasileiro de Sistemas Elétricos*
SLA	*Service Level Agreement*
SLO	*Service Level Objective*
SLR	*Systematic Literature Review (Revisão Sistemática da Literatura)*
SODAR	*Sound Detection and Ranging*
SQL	*Structured Query Language*
SQLC	*SQL Compiler (gerador de código a partir de consultas SQL)*
TLS	*Transport Layer Security*
TimescaleDB	*Extensão temporal para PostgreSQL*

TPC-DS	*Transaction Processing Performance Council – Decision Support*
UI	*User Interface (Interface do Usuário)*
UFMA	*Universidade Federal do Maranhão*
UML	*Unified Modeling Language*
UUID	*Universally Unique Identifier*
VM	*Virtual Machine (Máquina Virtual)*
VS Code	*Visual Studio Code*
ZIP	*Formato de arquivo compactado*

Lista de Símbolos

AUC	Área sob a curva ROC (classificador)
MAE	Erro absoluto médio
$MAPE$	Erro percentual absoluto médio [%]
ROC	*Receiver Operating Characteristic*
ZB	Zettabyte (unidade de volume de dados)
TJ	Terajoule (unidade de energia)
t_i	i-ésimo instante temporal em séries temporais
S_i	i-ésima série/sensor
SID	Identificador da série
V	Valor observado na série
z_{bottom}	Altura inferior de referência no cálculo de REWS [m]
z_{top}	Altura superior de referência no cálculo de REWS [m]
v_{bottom}	Velocidade do vento em z_{bottom} [m/s]
v_{top}	Velocidade do vento em z_{top} [m/s]
α	Expoente do perfil vertical do vento (lei de potência)
$REWS$	*Rotor Equivalent Wind Speed* [m/s]

Lista de Definições

Data Warehouse Repositório centralizado de dados estruturados e integrados, voltado à análise e relatórios, com infraestrutura mais rígida, maior custo por terabyte e otimizado para garantir desempenho, consistência e estabilidade das consultas analíticas.

Data Lakehouse Arquitetura que combina a flexibilidade e o menor custo de armazenagem de dados do Data Lake com a governança e o desempenho do Data Warehouse, operando sobre infraestrutura distribuída (cloud ou on-premise) para suportar grandes volumes de dados heterogêneos com crescimento elástico.

Streaming Processamento contínuo de dados gerados em tempo real ou quase real, permitindo ingestão, análise e resposta imediata a eventos, normalmente exigindo infraestrutura distribuída, escalável e de baixa latência.

Big Data Conjunto de tecnologias e práticas voltadas ao armazenamento, processamento e análise de volumes massivos de dados, variados em formato e gerados em alta velocidade, apoiado em arquiteturas distribuídas e escaláveis.

Cloud Storage Serviço de armazenamento de dados em infraestrutura remota e distribuída, acessível via rede, que oferece alta disponibilidade, elasticidade, durabilidade e cobrança conforme o uso.

Frameworks Estruturas conceituais que organizam ideias e conectam teoria a observações empíricas, atuando como boundary objects — conceitos ou artefatos suficientemente flexíveis e robustos para permitir comunicação e colaboração eficaz entre diferentes disciplinas ou grupos.

FIDLER Framework of Intelligent Data Lakehouse for Energy Resource, que utiliza a abordagem Data Lakehouse para armazenamento, análise e integração de dados em sistemas energéticos renováveis.

LIDAR Equipamento com laser infravermelho (Light Detection and Ranging) capaz de realizar medições atmosféricas e de altura, amplamente usado em projetos de energia eólica e estudos meteorológicos.

ADCP Acoustic Doppler Current Profiler, equipamento acústico capaz de medir correntes aquáticas (rios, estuários e oceano) em diferentes

profundidades, comum em projetos de energia de marés e pesquisas oceanográficas.

BI Business Intelligence, prática que combina aquisição, integração, análise e apresentação de dados para subsidiar decisões estratégicas.

ETL Processo Extract, Transform, Load que extrai dados de fontes diversas, transforma para atender regras de negócio e carrega em repositórios analíticos.

Boundary objects Entidades (conceituais ou físicas) que suportam a cooperação entre domínios distintos ao serem suficientemente abstratas para cada grupo, mas detalhadas o bastante para manter consistência.

Pipelines ETL Cadeias de etapas encadeadas que orquestram a extração, limpeza, transformação e carga de dados de forma automatizada e monitorada.

Vendor lock-in Situação em que uma dependência excessiva de um fornecedor específico limita a migração para outras tecnologias ou plataformas.

Schema Estrutura que define a organização dos dados, incluindo tabelas, campos, tipos e restrições que garantem consistência nos repositórios.

Diagramas UML Representações visuais da arquitetura e do comportamento de sistemas utilizando a Unified Modeling Language para facilitar entendimento e comunicação.

Structured data Dados organizados segundo um modelo predefinido (como tabelas relacionais ou colunas com tipos explícitos), o que torna simples consultas e validações.

Unstructured data Dados sem formato fixo ou esquema prévio, como textos livres, imagens e vídeos, exigindo técnicas específicas para indexação e análise.

RBAC Role-Based Access Control, modelo de controle de acesso em que permissões são atribuídas a papéis e os usuários herdam essas permissões conforme sua função.

ABAC Attribute-Based Access Control, modelo de autorização baseado em atributos dos usuários, recursos, ações e contexto.

AES-256 Advanced Encryption Standard com chave de 256 bits usado para cifrar dados sensíveis com segurança comprovada.

TLS Transport Layer Security, protocolo criptográfico que protege dados em trânsito por meio de autenticação e criptografia de canais.

Swimlanes — Faixas em diagramas de processo que organizam atividades por responsáveis, evidenciando fronteiras de responsabilidade em fluxos de trabalho.

Pipeline — Processo automatizado que ingere, transforma e move dados entre estágios (captura, preparação, armazenamento e consumo) dentro de uma arquitetura analítica.

Lista de Anexos

Sumário

1 INTRODUÇÃO

Este capítulo é apresentado o contexto, a problemática, a motivação, a hipótese, os objetivos, a justificativa, a solução proposta, a metodologia de pesquisa e a apresentação dos demais capítulos.

1.1 Contexto

Os *data centers* constituem instalações especializadas que concentram recursos computacionais e de armazenamento, operando de forma contínua para suportar serviços digitais essenciais, como computação e armazenamento de dados em nuvem, redes sociais, inteligência artificial (IA), *streaming* e plataformas corporativas. Essas estruturas formam a espinha dorsal da economia digital.

No entanto, seu elevado consumo energético, impulsionado pelo crescimento exponencial do tráfego global de internet — que aumentou mais de 20 vezes desde 2010 — e pela quadruplicação das cargas de trabalho entre 2015 e 2022, segundo a IEA (do inglês *International Energy Agency*) [14], suscita preocupações ambientais significativas. Esse cenário evidencia a urgência de estratégias que conciliem desempenho e sustentabilidade.

A intensificação da digitalização, resultante do aumento no número de usuários de internet — que mais que dobrou desde 2010 — e do crescimento de mais de 20 vezes no tráfego global de dados [14], ampliou de forma significativa a demanda por *data centers*. Em 2022, essas instalações e suas redes de transmissão consumiram entre 240 e 340 TWh, o que equivale a 1–1,3% da demanda elétrica global. Projeções da IEA indicam que esse consumo pode dobrar até 2026, devido ao avanço de tecnologias como IA generativa, *blockchain* e realidade aumentada.

Conforme a Figura 1.1 podemos observar esse crescimento exponencial da esfera global de dados, reforçando a pressão sobre os *data centers* e a necessidade de mitigar seu impacto energético.

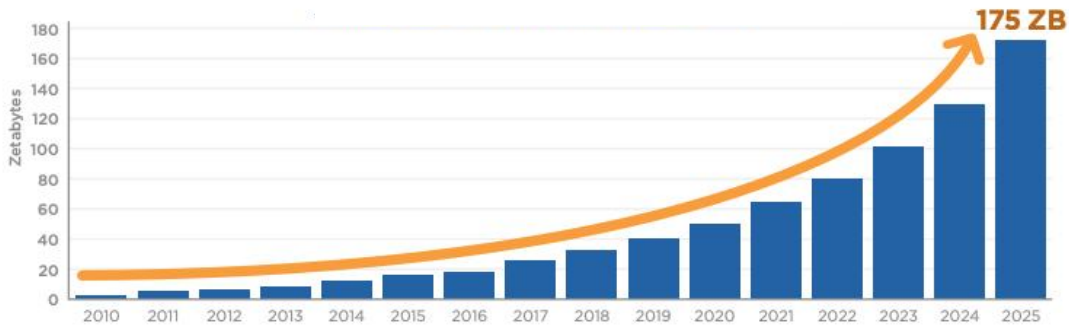


Figura 1.1 – Tamanho anual da esfera de dados em Zetabytes. Fonte: [1].

Com o crescimento acelerado no volume dos dados com o passar dos anos como visto na Figura 1.1, apesar dos avanços em eficiência energética, como a modernização de *hardware*, a refrigeração líquida e a migração para estruturas em nuvem otimizadas, o impacto ambiental dos *data centers* permanece significativo. O treinamento de modelos de *Deep Learning*, por exemplo, pode emitir até cinco vezes mais CO_2 do que um carro ao longo de sua vida útil [15].

Projeções otimistas sugerem uma possível estabilização das emissões até 2030 [16], mas o rápido aumento das cargas computacionais tende a superar esses ganhos no curto prazo. Esse desafio reforça a necessidade de soluções que equilibrem desempenho computacional e redução de emissões. O impacto ambiental dos *data centers* está diretamente ligado à matriz energética que os alimenta. Em 2022, fontes fósseis — como petróleo (30,2%), carvão (27,6%) e gás natural (23,1%) — representavam 81% da oferta energética global, enquanto fontes renováveis, como solar e eólica, respondiam por apenas 14% [2]. Podemos acompanhar a lentra transição da matriz energética global entre 1990 e 2022 analisando a Figura 1.2, que evidencia a dependência de combustíveis fósseis.

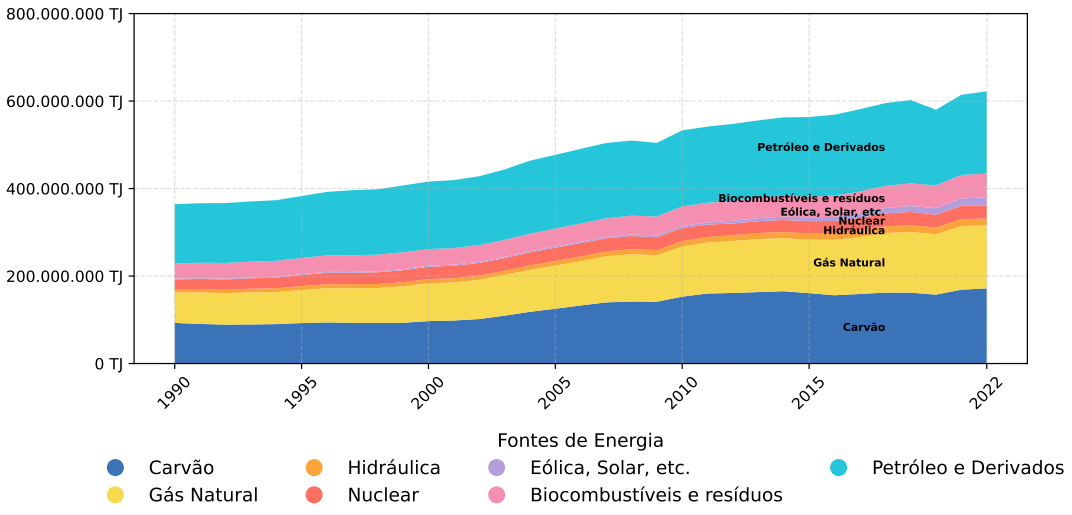


Figura 1.2 – Evolução da matriz energética mundial (1990–2022). Fonte: [2].

Essa predominância de fontes não renováveis dificulta a descarbonização do setor digital, demandando esforços coordenados para promover uma transição sustentável e diversificar a matriz energética. Para enfrentar esses desafios, grandes empresas de tecnologia, como AMAZON, GOOGLE, MICROSOFT e META, têm investido em contratos de compra de energia renovável (PPAs, do inglês *Power Purchase Agreements*), alcançando quase 50 GW [14]. No entanto, a variabilidade de fontes como solar e eólica, ilustrada na Figura 1.3, que detalha a composição da matriz energética global em 2022, limita sua capacidade de atender à demanda constante dos *data centers*. Essa limitação destaca a importância de uma matriz energética diversificada, que integre fontes renováveis com soluções de armazenamento e tecnologias complementares para garantir resiliência e sustentabilidade.

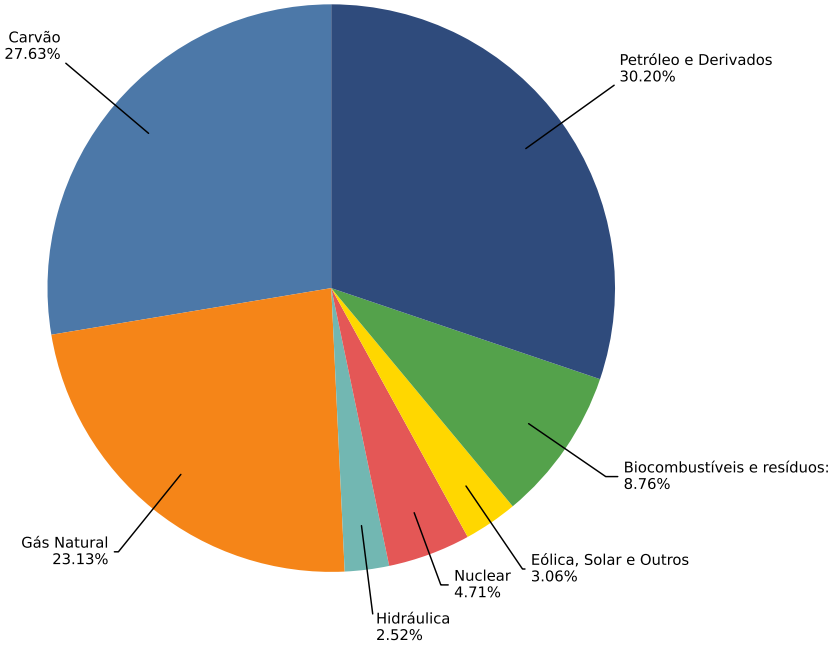


Figura 1.3 – Composição da matriz energética mundial (2022). Fonte: [2].

Nesse contexto global, o Brasil destaca-se por sua matriz energética, com 49% de fontes renováveis e 83,3% na matriz elétrica, majoritariamente hídrica [3]. Essa predominância hídrica, resultante de investimentos históricos desde os anos 1960 [17], contrasta com a média mundial, como ilustrado na Figura 1.4. Contudo, a dependência de hidrelétricas acarreta vulnerabilidades, como secas prolongadas e desafios de interconexão devido a extensas linhas de transmissão, o que incentiva a diversificação para a expansão sustentável de *data centers*.

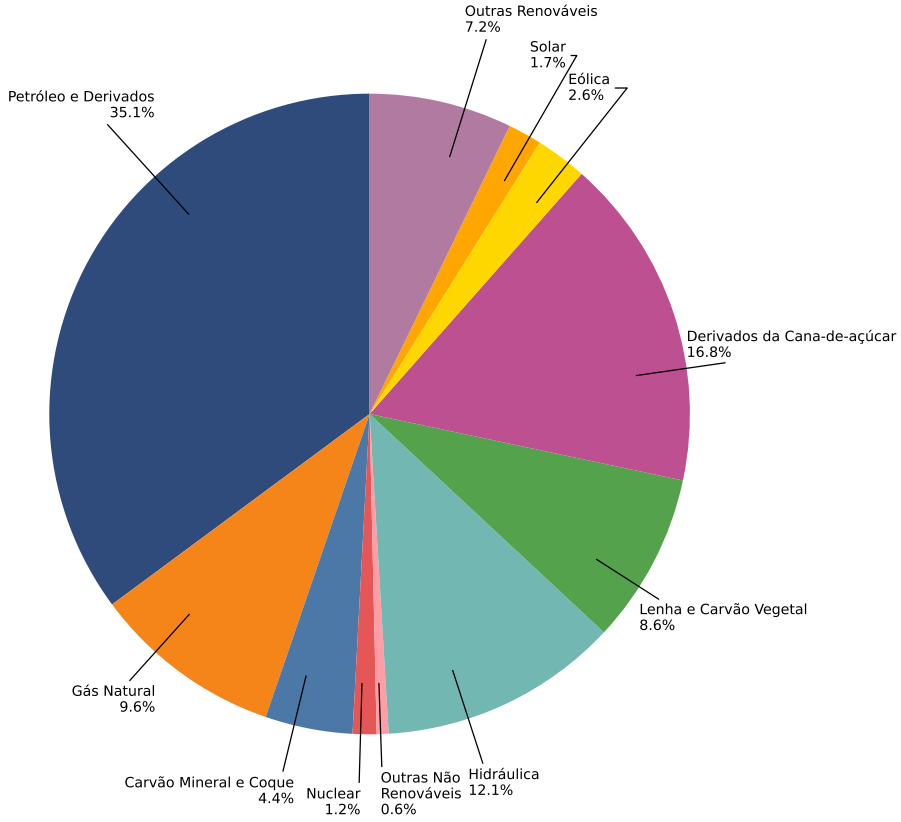


Figura 1.4 – Matriz energética brasileira (2023). Fonte: [3].

Podemos sintetizar e analisar esse contraste com base na Figura 1.5, o que evidencia a maior participação de renováveis no Brasil e as condições favoráveis para sua expansão.

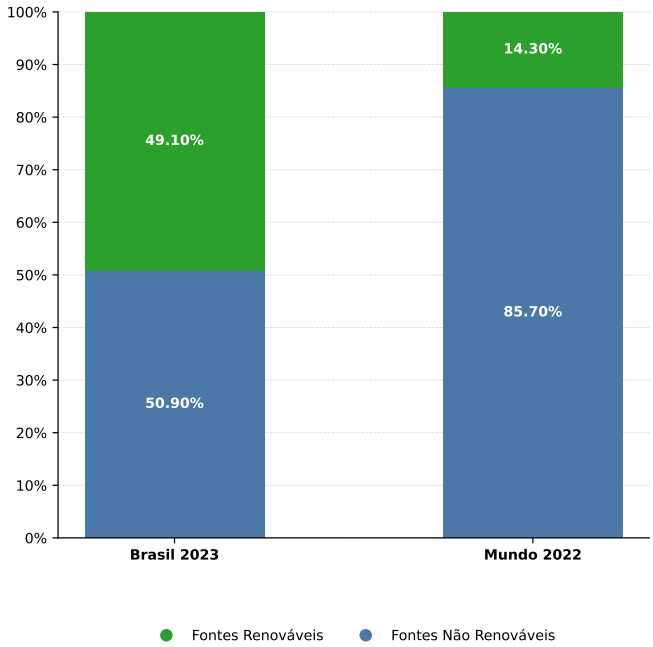


Figura 1.5 – Comparação entre fontes renováveis e não renováveis: Brasil (2023) e Mundo (2022). Fonte: [2, 3].

No contexto brasileiro, o Maranhão se posiciona como uma região estratégica para a expansão das fontes renováveis, em função de seu elevado potencial eólico e solar. Áreas como o Campo de Perizes apresentam fatores de capacidade eólica superiores a 0,70 em estudos preliminares baseados em *Rotor Equivalent Wind Speed* (REWS), associados a vantagens logísticas — proximidade da BR-135, presença de três linhas de transmissão, acesso ao Porto do Itaqui pelo Rio Mearim e conexão ferroviária — que favorecem a implantação de parques eólicos [18]. Iniciativas já em operação, como o Complexo Eólico Delta, da Serena Energia [19], apontam para a possibilidade de ampliar significativamente sua participação na matriz energética renovável do país.

A exploração dessas fontes renováveis depende de dados precisos para viabilizar projetos, como parques eólicos que utilizam tecnologias modernas de medição, como perfiladores do tipo LIDAR (do inglês *Light Detection and Ranging*) e SODAR (do inglês *Sound Detection And Ranging*), conforme nota técnica NT EPE-DEE-RE-057/2016-R4, que estabelece instruções para medições meteorológicas em parques eólicos da EPE (Empresa de Pesquisa Energética) [20].

Esses sensores, juntamente com dispositivos oceânicos como ADCP (do inglês *Acoustic Doppler Current Profile*), geram dados de correntes de maré — fundamentais para futuros projetos de turbinas de corrente de maré ou eólicas *offshore* —, produzindo volumes exponenciais de dados. Essa produção demanda soluções avançadas de armazenamento e análise em tempo real. Arquiteturas tradicionais, como *Big Data* e *Data Warehouses*, revelam-se inadequadas para lidar com o volume e a heterogeneidade desses dados.

Com essa finalidade, *frameworks* baseados na arquitetura *Data Lakehouse* mostram-se necessários para otimizar a integração, o armazenamento e a análise em tempo real de dados energéticos renováveis, promovendo maior escalabilidade, governança robusta e flexibilidade no gerenciamento de grandes volumes de dados heterogêneos. Essa abordagem favorece a sustentabilidade e a eficiência do setor digital em regiões estratégicas.

1.2 Problemática

Os *data centers* demandam energia constante e em escala crescente, com projeções de duplicação até 2026 [14]. No Brasil, onde a matriz elétrica é 83,3% renovável [3], atrair esses centros com energia limpa representa uma estratégia relevante. Contudo, essa atração depende da expansão acelerada de fontes eólica, maremotriz e *offshore* para complementar a base hídrica e garantir resiliência diante de secas e variações sazonais.

O principal obstáculo reside na identificação rápida, precisa e de baixo custo de zonas viáveis para novos parques. Investimentos em usinas renováveis exigem retorno garantido, com cada aplicação respaldada por dados confiáveis que comprovem viabilidade técnica

e econômica. As campanhas de medição — obrigatórias por no mínimo 12 consecutivos, conforme normas da EPE [20] — realizam-se com equipamentos isolados (LIDAR, SODAR, ADCP, torres micrometeorológicas) em regiões remotas, gerando volumes massivos de dados heterogêneos, frequentemente com lacunas decorrentes de falhas de conectividade, eventos sinistros ou interrupções operacionais.

Sistemas tradicionais (*Big Data*, *Data Warehouses*) não conseguem integrar dados brutos com dados sintéticos — estes gerados por inteligência artificial para reconstruir séries temporais interrompidas por falhas de equipamentos ou conectividade —, eles não realizam cálculos consolidados para avaliação do potencial energético. Métodos convencionais baseados na altura do *hub* (*hub height*) fornecem estimativas limitadas, enquanto o REWS oferece uma modelagem mais precisa ao considerar o vento médio sobre toda a área do disco rotor. Esse cálculo avançado possibilita testar virtualmente diferentes configurações de turbinas, identificar os modelos com maior eficiência para o perfil de vento local e quantificar com maior confiabilidade o potencial eólico da região [18]. Ademais, tais sistemas não fornecem relatórios auditáveis para investidores, o que pode levar a atrasos na validação de projetos, desperdício de recursos em campanhas inconclusivas e lentidão na escalabilidade renovável, conforme destacado em estudos sobre barreiras ao investimento em energia limpa [21].

No Maranhão, o Complexo Eólico Delta [19] demonstra viabilidade operacional, e áreas como Perizes apresentam REWS superior a 0,70 em estudos preliminares [18]. Replicar esse modelo em escala — incluindo potenciais projetos maremotrizes ou *offshore* — exige não apenas dados robustos, rápidos e confiáveis, mas também a transformação desses dados em informações estruturadas, auditáveis e apresentáveis. Isso inclui indicadores claros (como REWS, curvas de potência e estimativas de geração), relatórios automatizados e dashboards acessíveis, permitindo decisões de investimento com segurança e agilidade.

Diante disso, surgem questões centrais: como integrar e gerenciar dados brutos e sintéticos de sensores energéticos para viabilizar projetos eólicos e maremotrizes? Como desenvolver *frameworks* baseados em *Data Lakehouse* que assegurem interoperabilidade, escalabilidade e confiabilidade, promovendo a expansão de fontes renováveis e o desenvolvimento sustentável em regiões estratégicas?

1.3 Motivação

A convergência entre a expansão global de *data centers* e a transição energética sustentável cria um cenário estratégico para o Brasil. Grandes operadores, como Amazon, Google e Microsoft, buscam ativamente energia renovável por meio de PPAs para reduzir emissões de carbono [14], demandando fontes limpas, confiáveis e em escala. O Brasil,

com matriz elétrica majoritariamente renovável (83,3% em 2023 [3]), posiciona-se como candidato natural para atender essa demanda. No entanto, a histórica dependência de hidrelétricas [17] introduz vulnerabilidades climáticas que comprometem a resiliência necessária para alimentar cargas críticas de *data centers*.

Essa contradição — entre um potencial renovável excepcional e riscos de suprimento — revela a necessidade urgente de diversificar a matriz com fontes complementares, como eólica, solar e maremotriz. Essas fontes não atuam apenas como alternativas, mas como imperativos para garantir estabilidade operacional e atrair investimentos em infraestrutura digital sustentável.

O Maranhão apresenta um cenário favorável à expansão renovável, combinando viabilidade técnica (REWS $>$ 0,70 em Perizes [18]), exemplos operacionais bem-sucedidos como o Complexo Eólico Delta [19] e infraestrutura logística consolidada — fatores que reduzem barreiras de implantação.

No entanto, a ponte entre esse potencial mensurável e projetos executáveis é frágil. Expandir parques eólicos e maremotrizes exige investimentos decisivos e redução de risco técnico, mas as campanhas de medição seguem normas rigorosas (como a NT EPE-DEE-RE-057/2016-R4 para LIDAR/SODAR) e produzem dados atmosféricos e oceânicos volumosos, heterogêneos e com lacunas. Esses dados brutos não são decisões: é preciso integrá-los, qualificá-los e disponibilizá-los como informação útil (por exemplo, REWS ou potência estimada para uma região específica). Sistemas atuais falham justamente em transformar essas medições em confiança técnica e financeira, criando um gargalo na validação de zonas viáveis que:

- Retarda a validação de zonas viáveis para novos parques eólicos e maremotrizes;
- Aumenta incertezas técnicas e financeiras, desestimulando investidores;
- Compromete a velocidade necessária para aproveitar janelas de oportunidade no mercado.

Superar esse gargalo de dados constitui um desafio tecnológico fundamental para destravar o potencial renovável brasileiro. A transformação de dados brutos em confiança — com informações auditáveis, seguras e capazes de sustentar decisões de investimento — representa o cerne dessa motivação. Desenvolver um *framework* baseado em *Data Lakehouse* implica criar a base para converter medições incertas em certezas calculadas, campanhas demoradas em análises ágeis e potencial natural em projetos concretos e bancáveis.

Em essência, a motivação reside em estabelecer uma ponte entre medição e investimento, criando alicerces de confiança para que regiões como por exemplo o Maranhão capitalizem seus recursos na corrida global por energia limpa, auditável e estrategicamente confiável.

1.4 Hipótese da Pesquisa

Os dados gerados em sistemas energéticos renováveis, como os utilizados em projetos eólicos, solares e maremotrizes, apresentam características específicas do domínio. Incluem medições de sensores como perfiladores LIDAR, SODAR, torres micrometeorológicas e ADCP (velocidade do vento, incidência solar, direção, correntes oceânicas), dados meteorológicos (temperatura, pressão atmosférica) e informações sobre infraestrutura elétrica (potência, capacidade de geração). Esses dados, frequentemente heterogêneos e em grandes volumes, coletam-se em campanhas dispendiosas, realizadas em regiões de difícil acesso, conforme regulamentações da Empresa de Pesquisa Energética (EPE), que exigem pelo menos um ano de medições dependendo da tecnologia utilizada para garantir a viabilidade de projetos [20]. A diversidade de formatos e a complexidade de integração desses dados representam desafios significativos para a interoperabilidade e a análise eficiente [13].

Na prática, abordagens tradicionais para gerenciamento de dados, como arquiteturas de *Big Data* e *Data Warehouses*, revelam-se inadequadas para lidar com a heterogeneidade, o volume e a necessidade de análise em tempo real dos dados energéticos [15]. Essas soluções frequentemente carecem de flexibilidade para integrar dados brutos e sintéticos ou para suportar cálculos específicos, como o REWS, essencial para estimativas de potencial eólico [18]. Além disso, a ausência de governança robusta e a dificuldade em compensar falhas em medições — decorrentes de problemas de conectividade ou falhas de equipamentos — limitam a confiabilidade e a escalabilidade dos sistemas de suporte a projetos renováveis.

A arquitetura *Data Lakehouse*, que combina a flexibilidade dos *Data Lakes* com a governança dos *Data Warehouses*, emerge como solução promissora para superar essas limitações [5]. O *Framework of Intelligent Data Lakehouse for Energy Resource - FIDLER* adota *pipelines ETL* (Extração, Transformação e Carga) em Python sobre Amazon S3 (ou MinIO em ambientes privados), integrando dados heterogêneos e será definido em detalhes a seguir no trabalho (seção 2.4).

Em comparação a soluções ancoradas em serviços gerenciados como os do Azure [22] — que oferecem automação, elasticidade nativa e rapidez de implantação —, a escolha do FIDLER por *pipelines* em código aberto (Python) visa reduzir o *vendor lock-in*. Esse termo refere-se à dependência crítica de um único fornecedor, situação que eleva custos de migração e limita a flexibilidade arquitetural. A abordagem do FIDLER, portanto, privilegia a portabilidade, o controle fino sobre custos e ciclos de processamento, e maior transparência do código-fonte.

Portanto, a hipótese deste trabalho é que a adoção do *Framework* FIDLER, baseado na arquitetura *Data Lakehouse*, permite separar o gerenciamento de dados energéticos heterogêneos do processamento analítico e da lógica de decisão, promovendo reúso,

escalabilidade e confiabilidade. Essa abordagem facilita a integração de dados brutos e sintéticos, conforme normas da EPE [20], e suporta a tomada de decisão ágil por investidores e gestores. Assim, viabiliza a expansão de projetos eólicos e a operação sustentável de *data centers* alimentados por energia limpa, auxiliando na diversificação da matriz energética.

1.5 Solução Proposta

FIDLER, um *framework* baseado em *Data Lakehouse*, voltado à integração e análise de dados energéticos heterogêneos. O FIDLER adota uma estrutura conceitual utilizando o padrão *Model-View-Controller (MVC)* [23], que permite a separação entre a lógica de negócio, a manipulação de dados e a visualização dos resultados. A camada de modelo define as regras de negócio e as entidades relacionadas aos dados energéticos, enquanto a controladora coordena o fluxo de informações entre as camadas, e a camada de visualização apresenta os dados tratados. Dentro desse arcabouço, o FIDLER emprega técnicas típicas de arquiteturas *Data Lakehouse*, como a ingestão de dados brutos, o armazenamento em camadas, a transformação e padronização de formatos, o versionamento de dados e a disponibilização para consumo analítico.

O FIDLER trabalha com modelos abstratos dos equipamentos de medição utilizados em sistemas de energia renovável, como SODAR (perfiladores acústicos), LIDAR (perfiladores a laser), anemômetros 2D e 3D, piranômetros, ADCP e outros. Esses modelos são independentes de formato e origem dos dados, permitindo a representação unificada das medições e atributos associados. Um motor de transformação opera sobre esses modelos, convertendo-os em representações genéricas para armazenamento padronizado — uma vez que os padrões de dados brutos variam conforme o equipamento e o fabricante — e, posteriormente, em modelos especializados para aplicações analíticas específicas, como estimativas de potência eólica ou simulações de variabilidade temporal. Esse processo garante a compatibilidade entre dados de diferentes equipamentos e a reusabilidade dos fluxos de processamento em múltiplos contextos.

As operações de ingestão, transformação e análise organizam-se em um fluxo contínuo de etapas conhecido como *pipeline* de dados. Esse *pipeline* representa a sequência lógica de processamento que os dados percorrem, desde sua origem até o consumo final. No FIDLER, o *pipeline* é modular e adaptável, permitindo a integração contínua de múltiplas fontes de dados, com manutenção de metainformações temporais e geoespaciais ao longo do processo. As transformações executam-se de forma desacoplada da origem dos dados, facilitando a incorporação de novos sensores, a adaptação a campanhas distintas e a atualização de regras de negócio sem reconfiguração estrutural do sistema.

Embora o FIDLER ofereça mecanismos automatizados para ingestão, transformação e organização dos dados, a intervenção manual pode ser necessária em etapas como o mapeamento inicial de atributos, a definição de regras de negócio específicas e o ajuste de parâmetros analíticos — por exemplo, em equipamentos não mapeados previamente, como LIDARs, SODARs de diferentes fabricantes ou outros com a mesma finalidade. Além disso, as representações finais geradas a partir dos modelos abstratos consistem em estruturas prontas para análise, mas não incluem a implementação completa dos comportamentos de inferência, que devem ser desenvolvidos conforme a aplicação. Dessa forma, o FIDLER constitui um *framework* de apoio ao desenvolvimento de soluções baseadas em dados energéticos, com caráter semiautomático e adaptável a diferentes requisitos técnicos.

1.6 Justificativa

Soluções tradicionais de gerenciamento de dados, como os *Data Warehouses*, foram desenvolvidas para ambientes corporativos com dados estruturados, baixa variabilidade e atualizações periódicas. Embora eficazes em contextos convencionais, essas arquiteturas apresentam limitações significativas quando aplicadas a domínios com alta heterogeneidade e dinâmica de dados, como o setor energético renovável. A rigidez dos esquemas, os altos custos de manutenção e a dificuldade de escalar para diferentes tipos de dados comprometem sua utilidade em cenários que demandam flexibilidade e capacidade de integração com múltiplas fontes de dados simultâneas.

Ao mesmo tempo, observa-se um crescimento expressivo na demanda energética global, impulsionado principalmente pela digitalização de serviços, expansão dos dispositivos conectados e, sobretudo, pela ascensão de tecnologias de inteligência artificial. O treinamento e operação de modelos de IA consomem grandes volumes de energia e são suportados por uma infraestrutura massiva de *data centers*, cujo consumo elétrico estima-se entre 240 e 340 TWh ao ano, com tendência de crescimento acelerado [14]. Esse cenário impõe desafios adicionais à segurança energética e evidencia a importância de uma matriz diversificada, bem como a necessidade de arquiteturas computacionais mais sustentáveis e alinhadas com a transição energética.

Nesse contexto, a arquitetura *Data Lakehouse* tem ganhado crescente atenção na literatura técnica por sua capacidade de combinar a flexibilidade dos *Data Lakes* com a governança dos *Data Warehouses*, oferecendo uma alternativa moderna para o armazenamento, tratamento e análise de dados heterogêneos [5, 24–26]. Diversos estudos exploraram suas aplicações em domínios como computação biomédica [6], sustentabilidade industrial [27], governança de dados, metadados e catálogos [8] e otimização de consultas em larga escala [28]. No entanto, observa-se uma escassez de abordagens que proponham, avaliem ou implementem *frameworks* baseados em *Data Lakehouse* voltados especificamente

ao domínio das energias renováveis, especialmente considerando a integração de múltiplos sensores e a necessidade de fluxos analíticos contínuos e auditáveis. Essa lacuna na literatura técnica reforça a importância de investigações que adaptem e validem tais arquiteturas para os desafios inerentes ao setor energético.

Essa ausência representa uma lacuna relevante, considerando a complexidade dos sistemas de medição e monitoramento em projetos energéticos. Tecnologias como LIDAR, SODAR, ADCP e torres micrometeorológicas produzem grandes volumes de dados em diferentes formatos, resoluções e frequências. A heterogeneidade desses dados, aliada à necessidade de versionamento, rastreabilidade e análises preditivas em tempo quase real, demanda uma arquitetura robusta e flexível. Trabalhos que discutem plataformas de dados no setor energético ainda se baseiam em abordagens tradicionais de *ETL* e *warehousing* [29, 30], as quais não atendem integralmente às exigências técnicas desses ambientes.

Para enfrentar esses desafios, torna-se fundamental a organização dos dados por meio de fluxos analíticos estruturados que interligam as etapas de ingestão, transformação, validação (ETLs) e análise. Esses fluxos promovem automação, padronização e reuso de componentes, além de favorecerem a construção de processos auditáveis e escaláveis. Quando integrados a modelos de inteligência artificial, esses fluxos permitem a geração de *insights* em tempo real, desde a identificação de falhas até a previsão de padrões complexos de geração e consumo, elevando o grau de responsividade dos sistemas.

A proposta deste trabalho justifica-se pela necessidade de superar as limitações das arquiteturas convencionais frente à crescente complexidade e volume dos dados energéticos renováveis. O desenvolvimento de um *framework* especializado, que integra os princípios do *Data Lakehouse* com técnicas de engenharia de dados, fluxos analíticos e modelos de inteligência artificial, constitui uma contribuição original para o avanço da infraestrutura computacional voltada à transição energética. Ao articular flexibilidade, escalabilidade e inteligência, a abordagem proposta promove interoperabilidade e governança de dados em ambientes energéticos complexos, respondendo às exigências atuais de sustentabilidade, eficiência, robustez analítica e aumentar a produtividade em pesquisas ou deciões estratégica pelos usuários autorizados, que os utilizarão para finalidades analíticas específicas.

1.7 Objetivos

Este capítulo apresenta o objetivo geral e os objetivos específicos que orientam a realização desta dissertação. Os objetivos foram definidos com base na revisão da literatura, nas limitações técnicas identificadas em arquiteturas tradicionais de dados e na crescente demanda por soluções computacionais adaptadas à realidade dos sistemas de medição e

análise em energias renováveis.

1.7.1 Objetivo Geral

Propor e avaliar uma abordagem Lakehouse para organizar, integrar, transformar e analisar dados heterogêneos de campanhas renováveis, com automação, interoperabilidade, escalabilidade, governança e suporte a IA, validando com dados reais provenientes de campanhas de medição, com sensores como LIDAR, SODAR, ADCP e torres micrometeorológicas.

1.7.2 Objetivos Específicos

Os objetivos específicos são os seguintes:

1. Investigar os principais desafios relacionados à ingestão, padronização, armazenamento e análise de dados em projetos de energias renováveis, com foco na heterogeneidade de sensores, formatos e frequências de coleta;

2. Estudar e comparar arquiteturas modernas de dados, com ênfase nos modelos baseados em *Data Lakehouse*, destacando os mecanismos de versionamento, flexibilidade de estrutura de organização dos dados(*schema*), governança e suporte a análises em tempo real;

3. Definir um modelo conceitual genérico capaz de representar a estrutura comum dos dados energéticos renováveis, independentemente do tipo de equipamento ou sensor utilizado;

4. Derivar, a partir do modelo genérico, modelos específicos para os principais equipamentos de medição utilizados em campanhas energéticas, incluindo LIDAR, SODAR, ADCP e torres micrometeorológicas;

5. Propor a arquitetura MVC para a organização interna da API juntamente com a arquitetura de *Data Lakehouse* do *framework* FIDLER, incorporando o modelo conceitual, as camadas lógicas de armazenamento (*Bronze, Silver, Gold*), os fluxos analíticos (*data pipelines*), a aplicação de técnicas de inteligência artificial e as regras de negócio;

6. Implementar um protótipo funcional do *framework*, com módulos de ingestão, transformação, validação e visualização dos dados energéticos coletados, garantindo escalabilidade e rastreabilidade;

7. Avaliar a aplicabilidade do *framework* proposto por meio de sua aplicação a dados reais de campanhas de medição, testando sua capacidade de replicação, adaptação e robustez em diferentes contextos operacionais;

8. Validar o uso de inteligência artificial nos fluxos analíticos, com foco na predição de variáveis relevantes e reconstrução de dados para auxiliar o suporte à tomada de decisão em sistemas energéticos baseados em fontes renováveis.

1.8 Metodologia de Pesquisa

A metodologia de pesquisa utilizada neste trabalho descreve-se a seguir:

1. **Pesquisa bibliográfica** em livros, artigos de conferências, periódicos científicos, documentos técnicos, páginas institucionais e normas técnicas. O objetivo consistiu em levantar conceitos, fundamentos e o estado da arte nos seguintes temas:

   - Arquiteturas de dados com ênfase em *Data Lakehouse*;
   - Aplicações computacionais para energias renováveis;
   - Integração de perfiladores atmosféricos e oceânicos (LIDAR, SODAR, ADCP e torres micrometeorológicas);
   - Técnicas de engenharia de dados e fluxos analíticos;
   - Uso de inteligência artificial em sistemas energéticos;
   - Modelagem e padronização de dados energéticos com alta resolução temporal e espacial.

2. **Proposição do *framework* FIDLER** para organização, ingestão, transformação, análise de dados energéticos renováveis, reconstrução desses dados e aplicação de técnicas de inteligência artificial para previsões. Esta etapa divide-se nas seguintes subetapas:

   a) **Definição da arquitetura geral do *framework***. A arquitetura proposta integra duas estruturas complementares:

      - Uma **arquitetura de dados do tipo *Data Lakehouse***, organizada nas camadas de armazenamento *Bronze*, *Silver* e *Gold*, que define o fluxo, a governança e os processos analíticos dos dados energéticos.
      - Uma **API central desenvolvida segundo o padrão arquitetural MVC (Model-View-Controller)**, que gerencia a lógica de negócio, a interação com os dados e a exposição de serviços, operando sobre a estrutura do Lakehouse.

Esta integração dá suporte à ingestão automatizada, versionamento, fluxos analíticos (*pipelines*) e à aplicação de modelos de inteligência artificial.

b) **Modelagem dos dados energéticos**. Esta etapa contemplou:

- A definição de um modelo conceitual genérico para representar a estrutura comum dos dados de diferentes equipamentos;
- A derivação de modelos específicos para os principais perfiladores utilizados em campanhas de medição (LIDAR, SODAR, ADCP e torres micrometeorológicas);
- A estruturação de tabelas e entidades para armazenamento dos dados e metadados, com suporte a dados tabulares e temporais, utilizando extensões como *TimescaleDB*, *PostGIS* e *UUID-OSSP*.

c) **Implementação dos fluxos analíticos e integração com inteligência artificial**. Definiram-se:

- Os módulos de ingestão, transformação e validação dos dados;
- Os fluxos analíticos automatizados para análise contínua e escalável dos dados energéticos;
- A aplicação de técnicas de inteligência artificial voltadas à previsão de dados, reconstrução e suporte à tomada de decisão, através de relatórios gerados pelo sistema.

3. **Implementação do *framework* FIDLER** com ferramentas multiplataforma e foco em código aberto. Utilizaram-se VS Code em conjunto com Go (API e orquestração de dados), Python (análises e transformação), TypeScript com Vite (interface web) e SQLC (geração de código a partir de queries e manipulação de dados). Para o banco de dados, empregou-se o **migrate** (golang-migrate) para versionamento de esquema e execução de migrações SQL — garantindo que a evolução do modelo de dados seja reprodutível e rastreável em todos os ambientes. O projeto conteinerizou-se com Docker e orquestrou-se via Docker Compose, assegurando portabilidade, escalabilidade e replicabilidade do ambiente.

4. **Avaliação da abordagem proposta** por meio do desenvolvimento de exemplos ilustrativos com dados reais de campanhas de medição. Essa avaliação incluiu:

- Testes de ingestão, transformação e consulta de dados em diferentes formatos e resoluções temporais;
- Aplicação de modelos de inteligência artificial para predição de variáveis e/ou reconstrução de dados;
- Comparação com abordagens tradicionais de *Data Warehousing*, considerando aspectos de flexibilidade, desempenho e governança;

- Demonstração da adaptabilidade e escalabilidade do *framework*, avaliando sua capacidade de integração com novos sensores e contextos operacionais diversos.

5. **Comparação com trabalhos relacionados encontrados na literatura**, destacando as principais diferenças em termos de escopo, abordagem tecnológica, suporte à heterogeneidade de dados e uso de inteligência artificial. Essa etapa permitiu evidenciar as contribuições do *framework* FIDLER frente às limitações observadas nas soluções existentes.

1.9 Apresentação da Dissertação

Esta dissertação compõe-se de sete capítulos, dos quais os seis demais descrevem-se a seguir:

- O **Capítulo 2** apresenta a fundamentação teórica e tecnológica, discutindo os desafios de coleta e gerenciamento de dados em campanhas renováveis, a evolução das arquiteturas de dados até o *Data Lakehouse* e o uso de inteligência artificial para reconstrução de séries temporais;
- O **Capítulo 3** expõe o estado da arte por meio de uma revisão bibliométrica sistemática, destacando contribuições recentes, tendências e lacunas na aplicação de *lakehouse* a dados energéticos;
- O **Capítulo 4** descreve o desenvolvimento do *framework* FIDLER, com requisitos, decisões de arquitetura, metamodelos e fluxos operacionais orientados à governança e à reprodutibilidade;
- O **Capítulo 5** reúne exemplos ilustrativos da implementação do protótipo, incluindo ciclo de vida dos dados, organização por zonas no *lakehouse*, interfaces do sistema e documentação de API;
- O **Capítulo 6** analisa os resultados obtidos e posiciona o FIDLER frente aos trabalhos relacionados, com base nos critérios definidos no estado da arte;
- Por fim, o **Capítulo 7** apresenta as conclusões, contribuições, limitações e perspectivas de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo estabelece os fundamentos teóricos e tecnológicos que sustentam o desenvolvimento de uma plataforma integrada para gerenciamento, análise e reconstrução de dados em sistemas energéticos renováveis. Inicialmente, caracteriza-se o problema do gerenciamento de dados em campanhas de medição eólica e maremotriz, com ênfase nos desafios operacionais, logísticos e de qualidade impostos por ambientes remotos e pela natureza complexa das séries temporais geradas. Em seguida, traça-se a evolução histórica das arquiteturas de dados — desde os bancos relacionais até o paradigma *Data Lakehouse* —, contextualizando o surgimento do *Big Data* e os 5Vs como catalisadores dessa transformação. Apresentam-se os princípios de organização em camadas, governança e qualidade de dados, bem como as tecnologias computacionais selecionadas. Introduz-se o conceito de *framework* como estrutura integradora, alinhada ao *Data Lakehouse*. Por fim, fundamenta-se o emprego de inteligência artificial, com foco em *autoencoders*, para reconstrução de lacunas e geração de dados sintéticos que preservem padrões físicos e estatísticos.

2.1 Vulnerabilidades na Aquisição e Transmissão de Dados

A modernização das campanhas, com transmissão de dados em tempo real, introduziu uma camada de riscos operacionais e de segurança. A Tabela 2.1, adaptada de [13], detalha os principais riscos e as estratégias para mitigá-los.

Conforme a Tabela 2.1, a interrupção de transmissão em áreas de conectividade instável é uma causa direta de lacunas nas séries. Mais grave é a exposição a ameaças cibernéticas, como acesso não autorizado e sequestro de dados, que podem comprometer totalmente a integridade de uma campanha. As medidas de mitigação listadas exigem uma infraestrutura de TI robusta, muitas vezes ausente em implantações de campo.

2.2 Problemas na Padronização e Qualidade dos Dados

Para além da transmissão, a confiabilidade das análises é comprometida na origem. A Tabela 2.2, também adaptada de [13], categoriza os desafios inerentes à qualidade e padronização dos dados coletados.

Tabela 2.1 – Riscos e medidas de mitigação na aquisição e transmissão de dados em campanhas de medição. Fonte: Adaptado de [13].

Risco	**Descrição**	**Mitigação**
Interrupção de transmissão	Perda de conexão em regiões remotas, causando lacunas de dados.	Redundância em armazenamento local e uso de comunicação por satélite como *backup*.
Acesso não autorizado	Invasão de redes de dados e controle de sensores remotos.	Autenticação em dois fatores, *firewalls* e VPNs em todas as conexões remotas.
Sequestro ou adulteração de dados (ransomware)	Captura ou corrupção dos dados coletados, inviabilizando a série histórica.	Criptografia ponta a ponta e *backups* frequentes.
Exposição de dispositivos IoT	Vulnerabilidades em módulos de comunicação de sensores.	Atualizações regulares de *firmware* e monitoramento de tráfego de rede (IDS).

2.3 Evolução das Arquiteturas de Dados

A gestão de dados evoluiu em resposta às limitações de escala, flexibilidade e governança impostas pelo crescimento exponencial de volume, velocidade e variedade — os pilares do *Big Data*. O conceito de *Big Data*, formalizado na década de 2000, descreve conjuntos de dados cuja escala, dinamismo e complexidade desafiam sistemas tradicionais. Definido pelos 5Vs — *volume*, *velocidade*, *variedade*, *veracidade* e *valor* —, revela-se essencial para gerenciar dados de energias renováveis, onde sensores e fontes institucionais geram informações em diversos formatos e grandes quantidades [1, 31].

Os 5Vs podem ser contextualizados como segue:

- **Volume**: Sensores como LIDAR e ADCP geram terabytes de dados por campanha, exigindo armazenamento escalável e de baixo custo [1];
- **Velocidade**: Medições em tempo real (ex.: ventos a 10 Hz) requerem processamento rápido para detecção de falhas e ajustes operacionais [32];
- **Variedade**: Formatos como CSV, JSON, NetCDF, .dat e binários proprietários(dependendo do fabricante do equipamento utilizado para medição) dificultam integração e demandam *pipelines* de dados (sequências de etapas de processamento) flexíveis [33];

Tabela 2.2 – Problemas e soluções na padronização e qualidade dos dados coletados em campanhas de prospecção. Fonte: Adaptado de [13].

Problema	**Descrição**	**Solução Recomendada**
Heterogeneidade de Equipamentos	Diferentes fabricantes e tecnologias geram dados incompatíveis.	Utilizar sensores industriais certificados e validar as medições por meio de coletas paralelas.
Falta de Calibração e Manutenção	Equipamentos sem calibração periódica podem produzir leituras errôneas.	Implementar calibração regular conforme normas IEC e realizar manutenções preventivas.
Sensores Não Industriais	Equipamentos convencionais são menos resistentes e sujeitos a falhas em ambientes adversos.	Priorizar sensores e componentes industriais, projetados para condições severas.
Dados Incompletos ou Inconsistentes	Lacunas ou imprecisões dificultam simulações confiáveis.	Validar medições por redundância de sensores e aplicar filtros estatísticos para controle de qualidade.
Ausência de Normas Específicas	Áreas emergentes no Brasil ainda carecem de regulamentação específica.	Desenvolver padrões nacionais alinhados às diretrizes internacionais.

- **Veracidade**: Ruídos ambientais, falhas de sensores e lacunas demandam validação rigorosa para garantir confiabilidade em análises regulatórias [31];
- **Valor**: Extração de *insights* para previsão energética, manutenção preditiva e certificação EPE justifica investimentos em infraestrutura de dados [13].

Esses 5Vs caracterizam os dados de campanhas renováveis como um caso clássico de *Big Data*, impulsionando a transição para arquiteturas modernas. Os principais marcos evolutivos incluem:

1. **Bancos de Dados Relacionais (1970)**: baseados no modelo relacional de Codd [34], utilizam SQL e garantem propriedades *ACID*. São ideais para dados estruturados e transações, mas rígidos com esquemas variáveis, dados brutos ou em petabytes [30];

2. **Bancos de Dados Não Relacionais – NoSQL (2000–)**: surgiram para lidar com a *variedade* e *velocidade* do *Big Data*, oferecendo modelos flexíveis como chave-valor (Redis), documentos (MongoDB), colunar (Cassandra) e grafos (Neo4j). Sacrificam consistência forte em favor de escalabilidade horizontal e disponibilidade (teorema CAP), sendo adequados para logs, eventos em tempo real e dados semiestruturados [35]. Contudo, carecem de suporte nativo a consultas analíticas complexas e transações multi-registro;

3. ***Data Warehouses* (1990)**: focados em suporte à decisão, empregam modelagem dimensional (Kimball) ou corporativa (Inmon) com processos *ETL* (extração, transformação e carga) [29, 36]. Oferecem alto desempenho em agregações, mas exigem transformação prévia, duplicação de dados e esquemas fixos — inviáveis para séries temporais lacunares ou formatos proprietários;

4. ***Data Lakes* (2010)**: armazenam dados em formato nativo (*schema-on-read*) em *object storage* de baixo custo (ex.: S3, GCS). Habilitam experimentação, aprendizado de máquina e armazenamento de dados brutos, mas, sem governança, tornam-se *data swamps* com perda de rastreabilidade e qualidade [11];

5. ***Data Lakehouse* (2020–)**: paradigma híbrido que unifica a flexibilidade do *Data Lake* com a confiabilidade do *Data Warehouse*. Utiliza formatos abertos (Parquet, Delta Lake), transações *ACID*, versionamento, *time travel* e *engines* unificadas (Spark, Trino) [5, 24].

O *Data Lakehouse* elimina silos (repositórios de dados isolados por equipe ou sistema, com pouca interoperabilidade), reduz custos de movimentação e permite *workloads* simultâneos de BI, ETL e ML sobre a mesma base — ideal para dados de campanhas com lacunas, ruído e necessidade de reconstrução.

2.4 O Paradigma Data Lakehouse

O *Data Lakehouse* constitui um paradigma arquitetural que combina a flexibilidade e o baixo custo do *Data Lake* com a confiabilidade, governança e desempenho analítico do *Data Warehouse* [5]. Diferentemente do *Data Lake* tradicional — que armazena dados brutos em formato nativo com abordagem *schema-on-read* e sem garantias transacionais —, o *Lakehouse* introduz uma camada de metadados transacionais que habilita operações *ACID* (Atomicidade, Consistência, Isolamento, Durabilidade), versionamento, *time travel* e rastreamento de linhagem (*data lineage*), eliminando a necessidade de duplicação de dados entre sistemas [24]. Essa convergência sustenta-se em quatro componentes fundamentais:

- **Armazenamento em *object storage***: utiliza sistemas escaláveis como Amazon S3, MinIO ou serviços equivalentes como o Azure Blob Storage [22, 37], oferecendo durabilidade de 99,999999999% (11 9's) e custo reduzido por GB, ideal para volumes massivos de dados brutos de campanhas;
- **Formatos colunares abertos**: Parquet (formato aberto da Apache que armazena os dados por coluna, em vez de por linha, com compressão e esquema embutido) e ORC (formato aberto do ecossistema Hadoop que também armazena por coluna e mantém índices) permitem compactação eficiente, leitura seletiva de colunas e integração com *engines* analíticas, otimizando consultas sobre séries temporais de alta frequência;
- **Camadas de metadados transacionais**: *frameworks* como Delta Lake (cuja implementação é suportada no Azure como Azure Databricks Delta Lake [22]), Apache Iceberg ou Apache Hudi gerenciam concorrência, *snapshots* e otimizações automáticas (*compaction*, *z-ordering*), garantindo consistência em operações simultâneas de ETL e ML;
- ***Engines* de consulta unificadas**: Spark SQL (disponível no Azure HDInsight e Azure Databricks [22]), Trino ou Dremio executam consultas SQL ANSI diretamente sobre dados brutos com desempenho comparável a *data warehouses* tradicionais.

No contexto de campanhas de medição para energias renováveis, o *Data Lakehouse* viabiliza:

- Armazenamento fiel de dados brutos de LIDAR, SODAR e ADCP em formato nativo, sem transformação prévia;
- Integração fluida de dados sintéticos gerados por modelos de IA, com rastreamento automático de origem;
- Consultas analíticas unificadas sobre séries reconstruídas, sem duplicação entre camadas OLAP e OLTP;
- Conformidade com requisitos regulatórios da EPE por meio de auditabilidade completa e versionamento [20].

2.5 Organização em Camadas

A arquitetura em camadas — *bronze*, *silver* e *gold* — constitui uma prática consolidada em *Data Lakehouse* para estruturar o ciclo de vida dos dados, promovendo

reprodutibilidade, separação de responsabilidades e governança incremental [7]. Cada camada corresponde a um estágio de refinamento:

- **Camada Bronze**: representa o **estado bruto e imutável** dos dados, armazenados exatamente como recebidos (ex.: arquivos `.nc`, `.dat`, binários proprietários). Não há limpeza ou transformação; o foco reside na preservação histórica para auditoria e reprodutibilidade. Dados são particionados por campanha, data e sensor, com metadados mínimos (timestamp, hash de integridade, origem).
- **Camada Silver**: contém **dados validados, padronizados e enriquecidos**, prontos para análise. Inclui:
  - Normalização de unidades, fusos horários e esquemas;
  - Sinalização explícita de lacunas e *outliers*;
  - Integração de dados sintéticos com metadados de proveniência;
  - Versionamento transacional para rastrear transformações.

  Essa camada otimiza-se para consultas exploratórias e serve como base para *feature engineering*.
- **Camada Gold**: oferece **agregações de negócio, indicadores curados e *datasets* otimizados** para consumo final:
  - Cálculo de REWS, fator de capacidade, estatísticas de Weibull;
  - Materialização de agregações contínuas para *dashboards*;
  - *Datasets* preparados para treinamento de modelos preditivos.

  Dados são indexados, compactados e acessíveis via SQL ou APIs, com alto desempenho.

Essa estrutura elimina silos, reduz custos de movimentação e permite que diferentes perfis (engenheiros, analistas, cientistas de dados) operem em níveis apropriados de abstração [9].

2.6 Governança e Qualidade de Dados

A **governança de dados** constitui o conjunto de políticas, processos e tecnologias que asseguram a confiabilidade, segurança, conformidade e reutilização dos ativos de dados ao longo de seu ciclo de vida [6]. A **qualidade de dados**, por sua vez, avalia-se por dimensões como exatidão, completude, consistência, atualidade e conformidade com padrões regulatórios [8].

No *Data Lakehouse*, a governança implementa-se de forma nativa e abrange:

- **Controle de acesso granular**: políticas RBAC e ABAC definem permissões por usuário ou papel, impedindo alterações indevidas em dados *bronze*;
- **Criptografia**: AES-256 em repouso e TLS em trânsito, com gerenciamento de chaves centralizado;
- **Versionamento e *time travel***: cada alteração gera um *snapshot*, permitindo consultas a estados históricos para auditoria;
- **Linhagem de dados**: rastreamento automático do fluxo desde a ingestão até o consumo, essencial para validar reconstruções com IA;
- **Validação de qualidade**: regras automatizadas verificam:
  - Faixas físicas válidas (ex.: velocidade do vento entre 0 e 50 m/s);
  - Taxa de lacunas por altura de medição (< 10 %);
  - Conformidade com diretrizes da EPE (disponibilidade $>= 90$ %).
- **Catálogos de metadados**: registram informações técnicas (esquema, localização), de negócio (descrição, responsável) e científicas (campanha, metodologia), alinhando-se aos princípios *FAIR* [6].

Essa abordagem garante a integridade das análises, a rastreabilidade exigida por normas regulatórias e a confiança em simulações que orientam investimentos em infraestrutura renovável [9].

2.7 Conceito de Framework

Os *frameworks* constituem estruturas conceituais que organizam ideias, conectando teoria a observações empíricas. Eles atuam como *boundary objects* [4] – conceitos ou artefatos que, por serem suficientemente flexíveis e robustos, permitem a comunicação e colaboração eficaz entre diferentes disciplinas ou grupos de pesquisa, sem exigir um consenso completo. Um *meta-framework* proposto por [4] sistematiza o desenvolvimento dessas estruturas, descrevendo quatro processos mediadores: generalização empírica, adequação teórica, formulação de hipóteses e aplicação. Esses processos são fundamentados nos raciocínios lógicos de indução, dedução e abdução.

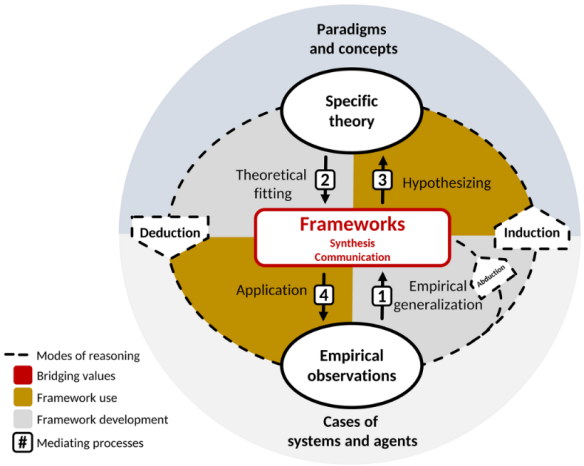


Figura 2.1 – *Meta-framework* para desenvolvimento de *frameworks*. Fonte: [4].

O *framework* baseado no paradigma *Data Lakehouse* integra dados de sensores (LIDAR, SODAR, ADCP e torres micrometeorológicas) e fontes institucionais (INEOF), promovendo *insights* para previsão energética, reconstrução de lacunas e colaboração científica. Alinha-se aos ODS 7 (Energia Acessível e Limpa), 9 (Indústria, Inovação e Infraestrutura) e 13 (Ação Contra a Mudança Global do Clima) [38], funcionando como um *boundary object* entre engenharia, ciência de dados e regulação.

2.8 Tecnologias Computacionais

O desenvolvimento da plataforma adota um conjunto de tecnologias modernas, maduras e de código aberto, selecionadas por sua robustez, escalabilidade e alinhamento com práticas contemporâneas de engenharia de software e ciência de dados. Todas as ferramentas empregam-se em suas versões estáveis mais recentes (até novembro de 2025), garantindo acesso aos últimos aprimoramentos de desempenho, segurança e compatibilidade. A seguir, detalham-se as principais tecnologias utilizadas, com ênfase em suas características técnicas e justificativas de escolha no contexto do projeto.

- **VS Code (Visual Studio Code, versão 1.95+)**: ambiente de desenvolvimento integrado (IDE) leve, extensível e multiplataforma, desenvolvido pela Microsoft. Suporta Go, Python, TypeScript e SQL com extensões oficiais (Go, Python, ESLint, PostgreSQL), oferecendo depuração integrada, controle de versão com Git, *terminal* embutido e integração com Docker. Sua arquitetura baseada em *plugins* permite personalização avançada, como *linting* em tempo real, autocompletar com IA (GitHub Copilot) e visualização de diagramas UML via extensões como *PlantUML*;

- **Go (Golang, versão 1.23+)**: linguagem compilada, de tipagem estática e com suporte nativo a concorrência via *goroutines* e *channels*. Utiliza-se para construção de serviços *backend* de alta performance, APIs RESTful e microsserviços. Sua eficiência em CPU e memória, combinada com compilação para binários *standalone*, torna-a ideal para processamento de grandes volumes de dados em ambientes com recursos limitados. Ferramentas como *golang-migrate* (v4.17+), *SQLC* (v1.26+) e *go-swagger* (v0.31+) integram-se para versionamento de esquemas, geração automática de código *type-safe* a partir de consultas SQL e documentação automática de APIs com suporte a OpenAPI 3.0 [39];

- **Python (versão 3.12+)**: linguagem interpretada de alto nível, com ecossistema rico para ciência de dados e aprendizado de máquina. Bibliotecas como *Pandas*, *NumPy*, *xarray*, *Scikit-learn* e *PyTorch* empregam-se para processamento de séries temporais, cálculo de indicadores energéticos (ex.: REWS, fator de capacidade) e treinamento de *autoencoders*. Sua flexibilidade permite integração com PostgreSQL via *psycopg3* e com S3/MinIO via *boto3*, além de suporte a *pipelines* de dados com *Prefect* ou *Airflow* quando necessário [40];

- **TypeScript (versão 5.6+) com Vite.js (versão 5.4+)**: para desenvolvimento de interfaces web modernas, reativas e de alta performance. O Vite.js oferece *build* ultrarrápido com *Hot Module Replacement* (HMR), suporte nativo a ES modules e otimização de *assets*, enquanto o TypeScript garante segurança de tipos em tempo de compilação. Frameworks como React (v18+) ou Svelte podem ser integrados, mas o foco reside em componentes reutilizáveis para visualização de séries temporais, mapas geoespaciais (via Leaflet ou Mapbox) e *dashboards* analíticos [41, 42];

- **PostgreSQL (versão 16+)**: sistema gerenciador de banco de dados relacional *open-source*, com suporte a transações *ACID*, índices avançados e extensibilidade via módulos. Três extensões revelam-se fundamentais:

  - *TimescaleDB (versão 2.15+)*: transforma o PostgreSQL em um banco otimizado para séries temporais, com *hypertables*, compressão automática, *continuous aggregates* e políticas de retenção. Essencial para armazenamento eficiente de milhões de registros por campanha;
  - *PostGIS (versão 3.4+)*: adiciona suporte a tipos geoespaciais (POINT, POLYGON). Utiliza-se para georreferenciamento de locais de medição, cálculo de proximidade entre sensores e integração com dados de rugosidade do terreno;
  - *uuid-ossp*: extensão nativa para geração de identificadores únicos universais (UUID v4), garantindo integridade referencial em ambientes distribuídos.

- **Docker (versão 27+) e Docker Compose (versão 2.29+)**: para conteinerização e orquestração de ambientes de desenvolvimento, teste e produção. Cada serviço

(API Go, *worker* Python, banco PostgreSQL, *frontend*, MinIO) encapsula-se em imagens leves (*multi-stage builds*), garantindo portabilidade, reproducibilidade e isolamento. O *Docker Compose* gerencia redes, volumes persistentes e dependências, facilitando a execução local com um único comando (`docker-compose up`);

- **MinIO (versão RELEASE.2025-11-01T09-30-00Z+)**: servidor de armazenamento de objetos compatível com a API do Amazon S3, implementado como um binário único em Go. Utiliza-se como camada *bronze* do *Data Lakehouse* em ambientes de desenvolvimento e testes, armazenando dados brutos em formato nativo (ex.: arquivos NetCDF, CSV proprietários, binários de LIDAR). Oferece alta performance, criptografia *server-side*, versionamento, *lifecycle policies* e suporte a políticas de acesso IAM. Em produção, pode ser substituído ou integrado ao Amazon S3 sem alterações no código, graças à compatibilidade total com a API S3.

2.8.1 Arquitetura de Software: Padrão MVC e Data Lakehouse

A plataforma segue o padrão arquitetural **Model-View-Controller (MVC)** adaptado ao contexto de aplicações *full-stack* com forte componente analítico:

- **Model**: camadas de acesso a dados (Go + SQLC para PostgreSQL, Python + Pandas para MinIO/S3), encapsulando lógica de persistência, validação e transformação. Integra o *Data Lakehouse* com camadas *bronze* (MinIO/S3), *silver* (TimescaleDB) e *gold* (agregações materializadas);
- **View**: interface web em TypeScript/Vite.js, com componentes reativos para visualização de séries, mapas, tabelas e relatórios;
- **Controller**: serviços RESTful em Go, orquestrando fluxos entre *frontend*, banco e *storage*, com *middlewares* para autenticação, *logging* e *rate limiting*. A documentação da API gera-se automaticamente com *go-swagger*, produzindo uma interface interativa em Swagger UI para testes de requisições HTTP.

Essa separação promove manutenibilidade, testabilidade e escalabilidade horizontal, revelando-se especialmente adequada para evoluções futuras com possíveis colaborações.

2.8.2 Modelagem com UML

A especificação do sistema documenta-se por meio de **diagramas UML (*Unified Modeling Language*, versão 2.5)**, seguindo o padrão estabelecido pela *Object Management Group* (OMG). Os diagramas abrangem diferentes perspectivas do sistema,

com ênfase no mapeamento das funcionalidades do *framework Data Lakehouse*. Os principais diagramas utilizados incluem:

- **Diagrama de Casos de Uso**: identifica atores (pesquisador, analista, administrador) e funcionalidades (ingestão de campanha, reconstrução de lacunas, geração de relatório EPE);
- **Diagrama de Classes**: modela entidades de domínio (Campanha, Sensor, SerieTemporal, Usuario), relacionamentos (1:N, N:N) e camadas de serviço;
- **Diagrama de Atividade**: detalha fluxos de trabalho, como ingestão na camada *bronze*, transformação para *silver* e agregação na *gold*;
- **Diagrama de Sequência**: representa interações em fluxos críticos, como *upload* de dados → validação → armazenamento → processamento;
- **Diagrama de Componentes**: ilustra a arquitetura em camadas do *Data Lakehouse* e integração entre PostgreSQL, MinIO/S3 e serviços externos;
- **Diagrama de Estados**: modela o ciclo de vida de uma série temporal, com transições entre estados como *Raw*, *Reconstructed* e *Aggregated*.

Esses diagramas garantem clareza na especificação, rastreabilidade de requisitos e base sólida para a implementação e futuras expansões do sistema. A combinação dessas tecnologias — todas em suas versões mais recentes — forma um ecossistema coeso, escalável e alinhado aos princípios do *Data Lakehouse*, MVC e engenharia de software moderna. O uso do MinIO como substituto local do S3, combinado com *go-swagger* para testes de API e Vite.js para *frontend*, permite desenvolvimento ágil, testes robustos e portabilidade entre ambientes locais e em nuvem, viabilizando o gerenciamento eficiente de dados heterogêneos, lacunares e em grande escala provenientes de campanhas de prospecção renovável.

2.9 Inteligência Artificial para Reconstrução e Geração de Dados Sintéticos

Lacunas em séries temporais comprometem a qualidade de análises regulatórias e simulações de longo prazo, especialmente quando se exige consistência estatística e coerência física. A literatura de reconstrução aponta os *autoencoders* como uma família adequada para esse cenário, por aprenderem representações compactas capazes de preservar estruturas não lineares durante a reconstrução [43]. Em contextos com falta de amostras e heterogeneidade, variações condicionais e adversariais também têm sido empregadas com

bons resultados na recomposição de sinais [44]. Além disso, trabalhos clássicos de redução de dimensionalidade via *autoencoder* reforçam o papel desses modelos como base para reconstrução estável de dados [45]. Esses resultados são geralmente organizados em três aplicações principais:

- **Imputação contextual**: reconstrução de valores ausentes com base em padrões aprendidos [43];
- **Geração de dados sintéticos**: preenchimento consistente de lacunas a partir do modelo treinado [44];
- **Detecção de anomalias**: identificação de desvios a partir dos resíduos de reconstrução [45].

Do ponto de vista teórico, essa linha de pesquisa também fundamenta o uso de representações latentes como suporte a tarefas de previsão, uma vez que a mesma estrutura que aprende padrões temporais úteis para reconstrução pode ser estendida à modelagem preditiva [43,44]. Assim, os *autoencoders* oferecem um arcabouço unificado para reconstrução e previsão, mantendo compatibilidade com diferentes níveis de complexidade e com futuras extensões arquiteturais.

2.10 Síntese

Este capítulo delineou o contexto e os desafios das campanhas de medição em ambientes remotos, destacando problemas de transmissão, heterogeneidade e lacunas nas séries temporais. Em seguida, apresentou a evolução das arquiteturas de dados e justificou a adoção do paradigma *Data Lakehouse*, com organização em camadas, governança e qualidade como pilares para dados energéticos. Também fundamentou o conceito de *framework* como estrutura integradora entre tecnologia, processos e objetivos científicos. Por fim, descreveu o conjunto de tecnologias adotadas, priorizando soluções gratuitas e de código aberto no protótipo, e sustentou teoricamente o uso de *autoencoders* como base para reconstrução de lacunas e geração de dados sintéticos. Esses elementos estabelecem a base conceitual que orienta a arquitetura e a implementação apresentadas nos capítulos seguintes.

3 ESTADO DA ARTE

Este capítulo organiza o estado da arte sobre *Data Lakehouse* e arquiteturas relacionadas, com ênfase em integração de cargas, governança, formatos abertos e suporte a séries temporais. A revisão bibliométrica sistematiza a produção recente e destaca tendências, contribuições recorrentes e limitações reportadas, incluindo desafios de maturidade tecnológica e de aplicação a dados heterogêneos. A seguir, apresenta-se o processo de revisão bibliométrica e a seleção dos artigos utilizados.

3.1 Revisão Bibliométrica

Uma revisão sistemática da literatura (*SLR*), inspirada em [46], foi conduzida para mapear tendências em arquiteturas *Data Lakehouse* e sua aplicação em energias renováveis. A metodologia seguiu três etapas:

- **Estratégia de Busca**: Artigos publicados entre 2019 e 2025 foram coletados nas bases *IEEE Xplore*, *ACM Digital Library*, *SpringerLink* e *ScienceDirect*, utilizando palavras-chave como *Data Lakehouse*, *renewable energy*, *wind energy*, *tidal energy*, *time series*, *big data analytics* e *data governance*. Critérios de inclusão abrangeram artigos revisados por pares, relatórios técnicos e *preprints* com relevância prática. Critérios de exclusão eliminaram estudos duplicados, sem aplicação tecnológica ou fora do período;
- **Análise Bibliométrica**: O software *VOSviewer* (versão 1.6.20) gerou mapas de coocorrência de palavras-chave com base em títulos, resumos e palavras-chave autorais. Foi aplicada normalização por força de associação, com limiar mínimo de 3 ocorrências por termo;
- **Análise Qualitativa**: Leitura integral dos artigos selecionados identificou padrões temáticos, arquiteturas propostas, tecnologias empregadas e lacunas, especialmente na integração entre *Data Lakehouse* e séries temporais de alta frequência em contextos renováveis.

A busca resultou em 35 artigos relevantes, conforme detalhado na Tabela 3.1.

A Tabela 3.2 revela que apenas 12 artigos mencionam *Data Lakehouse* no título, indicando que o conceito é emergente e frequentemente abordado como parte de arquiteturas mais amplas de dados.

Tabela 3.1 – Número de artigos relevantes por base de dados (2019–2025).

Base de Dados	Número de Artigos
SpringerLink	10
ACM Digital Library	10
IEEE Xplore	8
Elsevier (ScienceDirect)	7
Total	35

Tabela 3.2 – Artigos com *Data Lakehouse* no título (2019–2025).

Base de Dados	Número de Artigos
SpringerLink	4
IEEE Xplore	4
ACM Digital Library	3
Elsevier (ScienceDirect)	1
Total	12

A Figura 3.1 apresenta a rede de coocorrência de palavras-chave, destacando clusters centrais em torno de *Data Lakehouse*, *big data*, *analytics*, *cloud computing* e *machine learning*. Há conexões periféricas com *renewable energy*, *time series* e *wind power*, mas pouca interseção direta, evidenciando uma lacuna na aplicação prática do *Lakehouse* em dados de sensoriamento renovável.

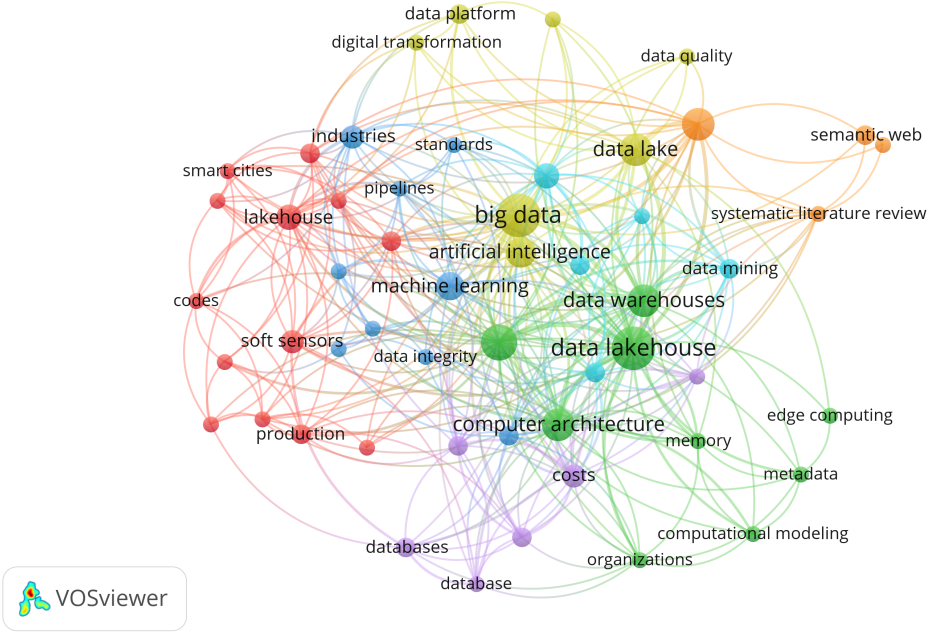


Figura 3.1 – Mapa de coocorrência de palavras-chave gerado pelo *VOSviewer*.

A Figura 3.1 indica que nós maiores correspondem a termos com maior frequência e que a espessura das linhas reflete a força de coocorrência.

3.2 Contribuições dos Principais Autores

A análise qualitativa dos artigos selecionados permitiu identificar um núcleo de autores cujas propostas definem os contornos técnicos e conceituais do *Data Lakehouse* moderno. Esta seção detalha essas estruturas, organizando-as por foco técnico predominante e analisando sua arquitetura, tecnologias propostas, vantagens e limitações. Para enriquecer a apresentação, são incluídas figuras representativas extraídas dos artigos originais e diagramas UML desenvolvidos como parte deste trabalho.

3.2.1 Integração e Unificação de *Workloads*

Nesta subseção, apresenta-se a contribuição sobre a unificação de cargas de trabalho no *Lakehouse*.

- **Armbrust et al. (2021)** [5]: Apresentam o conceito inicial de *Data Lakehouse*, propondo uma unificação arquitetural entre *Data Lakes* e *Data Warehouses*. A arquitetura fundamenta-se em três pilares: formatos de armazenamento abertos (*Apache Parquet*), uma camada de metadados transacional (*Delta Lake*) e suporte

a múltiplos *engines* de processamento (*Spark*, *Presto*). A Figura 3.2 é apresentada para ilustrar a evolução das arquiteturas de dados até o modelo *Lakehouse*.

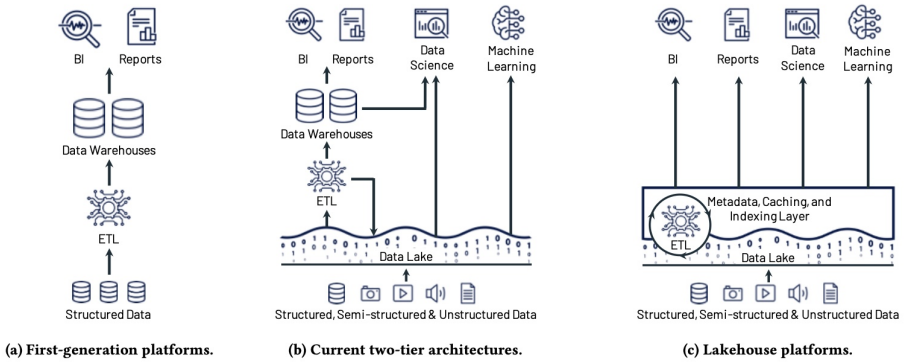


Figura 3.2 – Evolução das arquiteturas de dados para o modelo *Lakehouse*. ETL (Extrair, Transformar e Carregar) organiza e prepara dados brutos, enquanto BI (Business Intelligence) os transforma em insights para tomada de decisões estratégicas [5].

A Figura 3.2 ilustra a transição para o modelo unificado. O *Delta Log* garante transações ACID e versionamento, permitindo que cargas de BI, ML e ETL coexistam sobre o mesmo repositório.

Vantagem: Elimina duplicação de dados e permite análise direta sobre dados brutos em escala. *Limitação*: Dependência de infraestrutura distribuída e alta complexidade operacional.

3.2.2 Governança, FAIR e Reutilização Científica

Em seguida, resume-se a contribuição que integra governança e princípios *FAIR* ao ciclo de vida do *Lakehouse*.

- **Beg et al. (2021)** [6]: Propõem a integração dos princípios *FAIR* ao ciclo de vida do *Data Lakehouse*. Seu *framework* estrutura os dados em camadas *bronze*, *silver*, *gold* com catálogo ativo de metadados e linhagem de dados. A Figura 3.3 é apresentada para sintetizar o fluxo de ingestão, preparação e análise no *Lakehouse*.

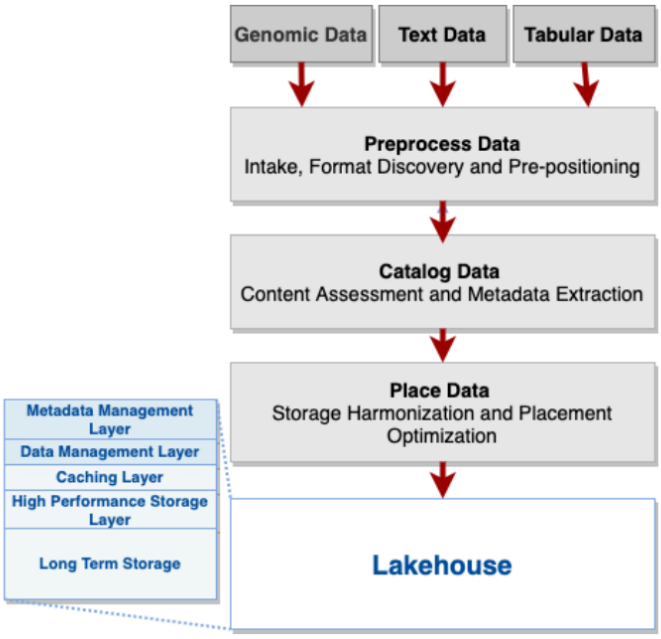


Figura 3.3 – Fluxo de ingestão, preparação e análise em ambiente *Lakehouse* [6].

A Figura 3.3 mostra o fluxo arquitetural. O modelo defende *ELT* sobre *ETL* e utiliza identificadores persistentes (PIDs) para reprodutibilidade. *Vantagem*: Promove reprodutibilidade científica e conformidade regulatória. *Limitação*: Exige catalogação intensiva e ferramentas maduras.

3.2.3 Organização Modular e Curadoria por Camadas

Nesta parte, apresenta-se a contribuição que formaliza a organização em camadas e sua curadoria.

- **Janssen (2022)** [7]: Formaliza o modelo de organização modular em camadas *bronze-silver-gold* como padrão para governança prática.

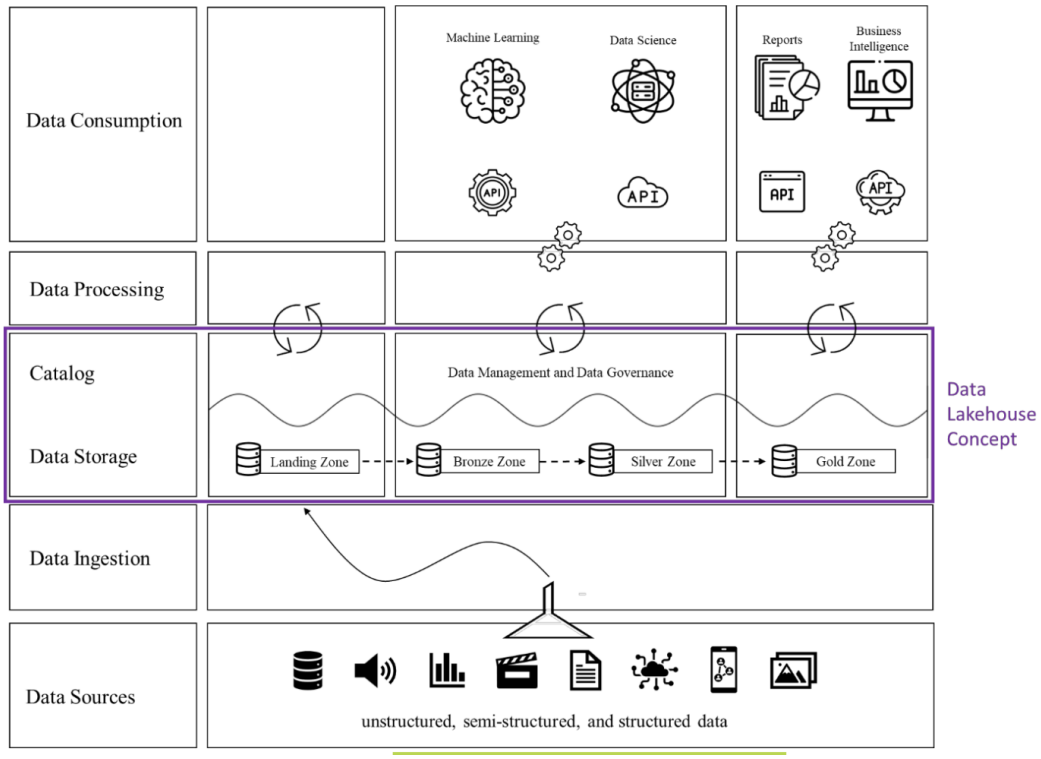


Figura 3.4 – Arquitetura modular de referência para *Data Lakehouse* [7].

A Figura 3.4 detalha a divisão zonada com papéis e fluxos definidos. A estrutura separa dados brutos, curados e prontos para análise. *Vantagem*: Simplicidade conceitual, escalabilidade e facilita colaboração. *Limitação*: Possível acoplamento a ecossistemas específicos.

3.2.4 Escalabilidade e Separação de Responsabilidades

Neste tópico, apresenta-se a contribuição que separa armazenamento e computação para escalar custos e desempenho.

- **Kumar et al. (2022)** [47]: Defendem a separação radical entre armazenamento (ex.: S3, GCS) e computação (ex.: Spark, Trino). O modelo otimiza custos em nuvem através de escalabilidade elástica.

  Vantagem: Maximiza eficiência de custos e elasticidade.

  Limitação: Latência em ambientes com conectividade limitada.

3.2.5 Governança Ativa e Federada

Aqui, apresenta-se a contribuição sobre governança federada voltada a ambientes interinstitucionais.

- **Rain et al. (2024)** [8]: Propõem um modelo de governança federada para ambientes interinstitucionais, com catálogo central de metadados.

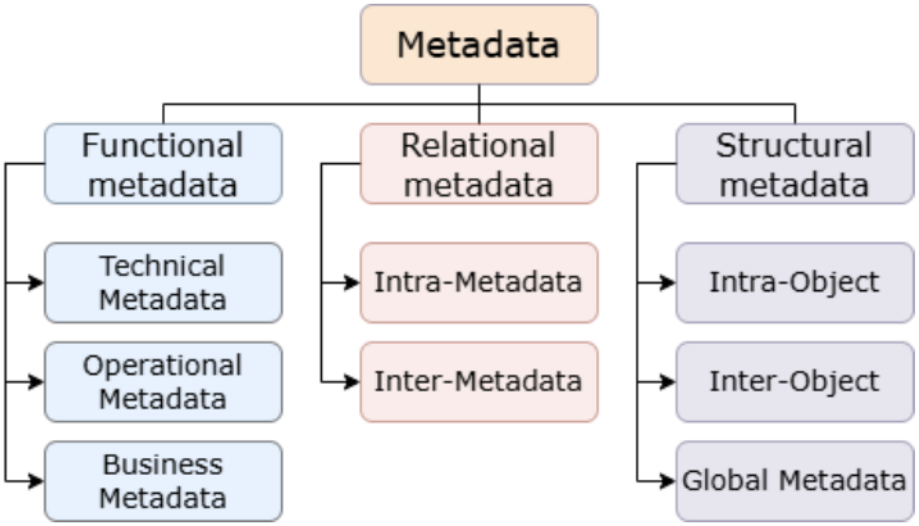


Figura 3.5 – Classificação dos metadados em técnico, operacional e semântico [8].

A Figura 3.5 mostra a estrutura hierárquica de metadados. Inclui controle de acesso RBAC/ABAC e *logs* de auditoria. *Vantagem*: Facilita colaboração segura entre instituições e evita *data swamps*. *Limitação*: Complexidade na definição de regras para dados heterogêneos.

3.2.6 Otimização para Séries Temporais

Nesta seção, apresenta-se a contribuição que adapta o *Lakehouse* a séries temporais de alta frequência.

- **Pohl et al. (2024)** [9]: Propõem extensões ao *Lakehouse* para séries temporais de alta frequência, usando bancos temporais como *TimescaleDB*. A Figura 3.6 introduz a arquitetura conceitual proposta.

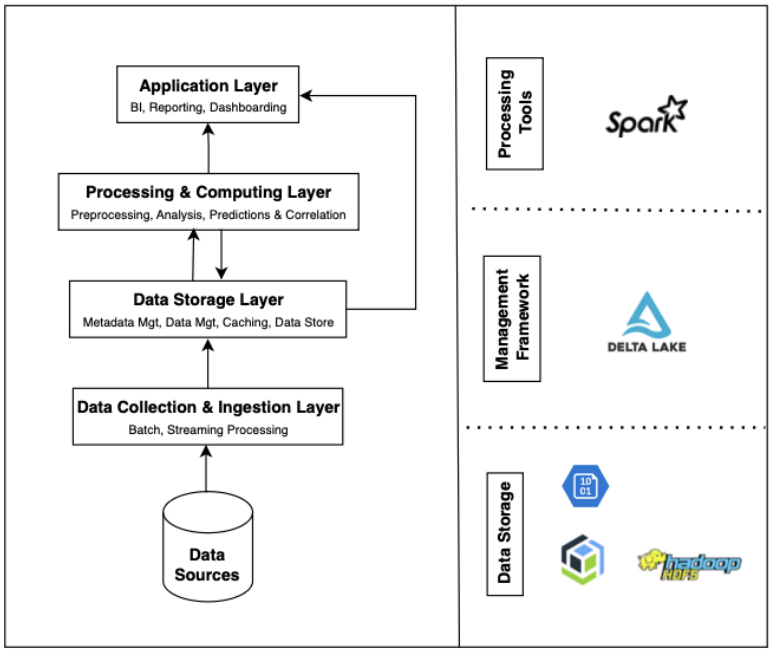


Figura 3.6 – Arquitetura conceitual para séries temporais energéticas [9].

A Figura 3.6 apresenta a arquitetura com integração de ML preditivo. Inclui técnicas de *resampling* e compactação temporal. *Vantagem*: Alta performance em consultas temporais complexas. *Limitação*: Foco restrito, menos ênfase em governança geral.

3.2.7 Compactação e Eficiência

Neste ponto, apresenta-se a contribuição voltada a técnicas de compactação e eficiência de armazenamento.

- **Sun et al. (2024)** [11]: Focam em técnicas avançadas de compactação (como o uso de *Parquet* com *Dictionary Encoding*) para maximizar a redução do espaço de armazenamento.
  - *Vantagem*: Redução de até 80% no espaço utilizado.
  - *Limitação*: Introduz *overhead* no processo de escrita e pode exigir reprocessamento para consultas complexas.

3.2.8 Flexibilidade e Multi-format

Por fim, apresenta-se a contribuição que discute a coexistência de múltiplos formatos de tabela no *Lakehouse*.

- **Schneider et al. (2023)** [24]: Propõem a adoção e coexistência de múltiplos *table formats* (como *Delta Lake*, *Apache Iceberg* e *Apache Hudi*) dentro de um mesmo ambiente de *Lakehouse*.
  - *Vantagem*: Maior adaptabilidade a diferentes casos de uso e *stacks* tecnológicos existentes.
  - *Limitação*: Risco de fragmentação de dados e aumento significativo na complexidade de gerenciamento.

Para consolidar a visão dos principais autores sobre as dimensões de um *Data Lakehouse*, a Tabela 3.3 apresenta uma síntese comparativa, destacando o foco principal, as vantagens e as limitações de cada abordagem estudada.

Tabela 3.3 – Síntese comparativa das abordagens propostas para a arquitetura de *Data Lakehouse*.

Autor (Ano)	**Foco Principal**	**Vantagens**	**Limitações / Desafios**
[5]	Integração Arquitetural	Unificação de cargas de trabalho (BI, ML, ETL); Análise em grande escala.	Alta complexidade operacional inicial.
[6]	Governança (FAIR)	Promove reprodutibilidade científica e rastreabilidade dos dados.	Requer catalogação meticulosa e contínua.
[7]	Organização Modular	Simplicidade na implementação; Escalabilidade horizontal.	Forte dependência do ecossistema de ferramentas adotado.
[47]	Escalabilidade em Nuvem	Custo-efetividade e elasticidade em ambientes cloud.	Desempenho pode ser limitado em cenários *offline* ou híbridos.
[8]	Governança Federada	Facilita colaboração segura entre domínios; Auditabilidade robusta.	Complexidade na definição e manutenção de políticas de acesso.
[9]	Otimização para Séries Temporais	Desempenho superior para consultas temporais e sequenciais.	Oferece menos suporte nativo a mecanismos abrangentes de governança.
[11]	Eficiência (Compactação)	Redução drástica (até 80%) no espaço de armazenamento.	*Overhead* computacional na escrita; pode necessitar de reprocessamento.
[24]	Flexibilidade (Multi-format)	Adaptabilidade a diversos casos de uso e *stacks* tecnológicos.	Risco de fragmentação de dados e aumento da complexidade de gestão.

A análise comparativa resume a natureza multifacetada e em evolução da arquitetura *Lakehouse*. Conforme evidenciado na Tabela 3.3, não existe uma solução única ou universal. Cada abordagem otimiza uma dimensão específica — como integração, governança, desempenho, custo ou flexibilidade — frequentemente em detrimento de outras. Portanto, a implementação de um *Data Lakehouse* eficaz depende de uma escolha arquitetural consciente, que priorize os atributos mais críticos para o contexto organizacional e técnico em questão, equilibrando as vantagens prometidas com os desafios operacionais inerentes a cada opção.

3.2.9 Complementos visuais e padrões recorrentes

Para aproveitar o acervo de figuras já extraídas dos artigos, destacam-se a seguir recortes adicionais que reforçam evolução arquitetural, papéis de acesso, formatos e esquemas de séries temporais.

Em Armbrust et al. [5], os recortes a seguir detalham elementos centrais do *Lakehouse*. A Figura 3.7 apresenta o desenho de sistema com camadas de acesso (APIs SQL e *DataFrame*), uma camada de metadados (caching e indexação) e o *data lake* em formatos abertos (por exemplo, Parquet).

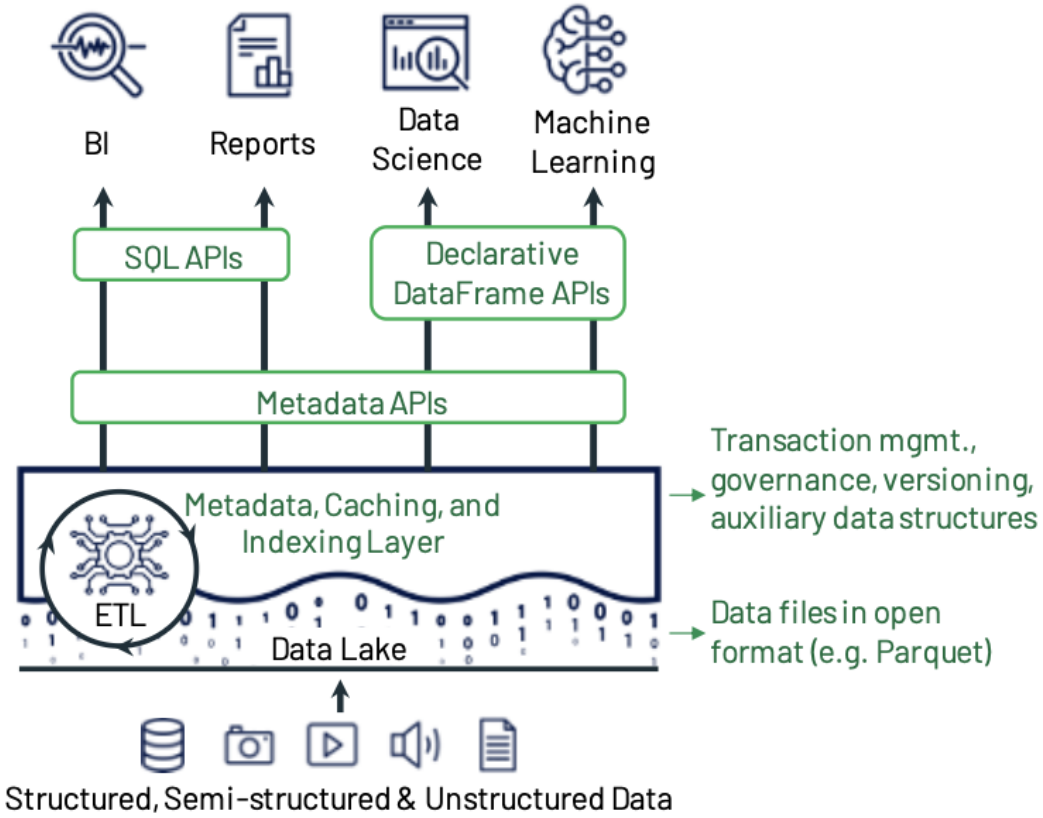


Figura 3.7 – Detalhamento da camada transacional e do *Delta Log* no *Lakehouse* [5].

O esquema contextualiza a integração de cargas heterogêneas (BI, relatórios, ciência de dados e ML) sobre o mesmo repositório, com governança e versionamento concentrados na camada de metadados.

Na sequência, a Figura 3.8 sintetiza resultados de desempenho do *TPC-DS*, comparando duração e custo do teste entre diferentes *data warehouses* e o *Delta Engine* (em modos *on-demand* e *spot*).

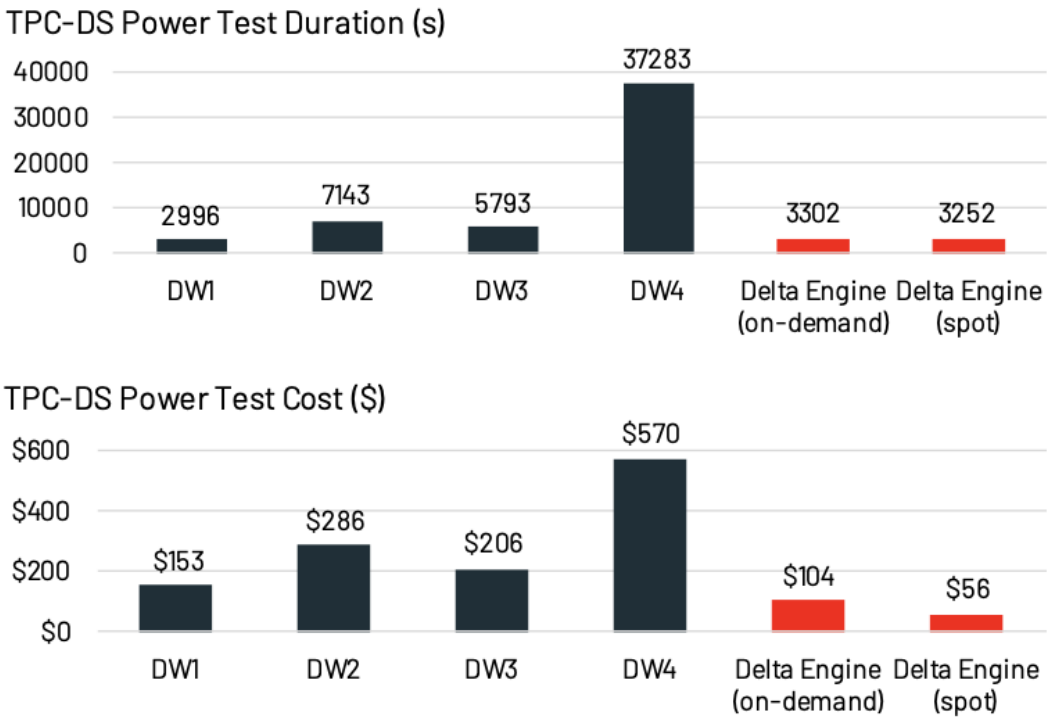


Figura 3.8 – Fluxos de leitura/escrita otimizados em formato aberto [5].

O gráfico evidencia ganhos relevantes de tempo e custo em relação às alternativas comparadas, reforçando a viabilidade do *Lakehouse* para análises em escala.

Em Beg et al. [6], a Figura 3.9 é introduzida para explicitar a estrutura de governança proposta.

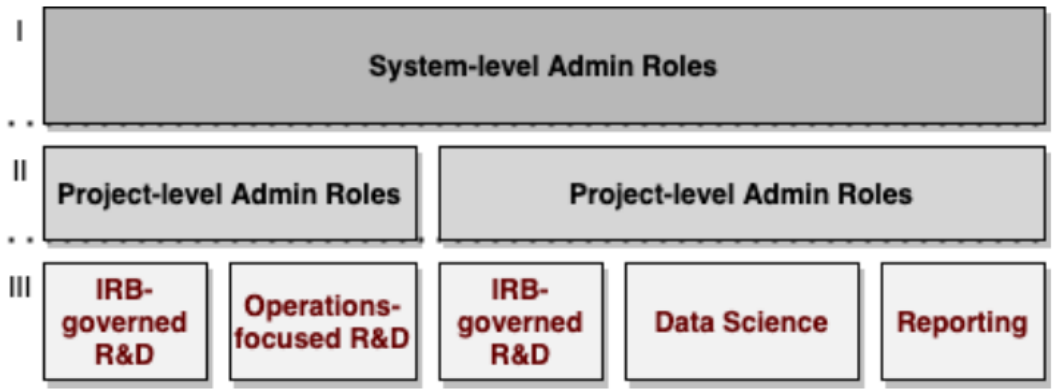


Figura 3.9 – Papéis de acesso e governança em ambientes *Lakehouse* [6].

A Figura 3.9 organiza a governança em níveis, distinguindo papéis administrativos de sistema e de projeto, além de perfis de uso (P&D, ciência de dados e relatórios).

Essa hierarquia destaca a necessidade de separação de responsabilidades para garantir conformidade e rastreabilidade no *Lakehouse*.

Na sequência, em Beg et al. [6], a Figura 3.10 introduz uma comparação entre formatos de armazenamento.

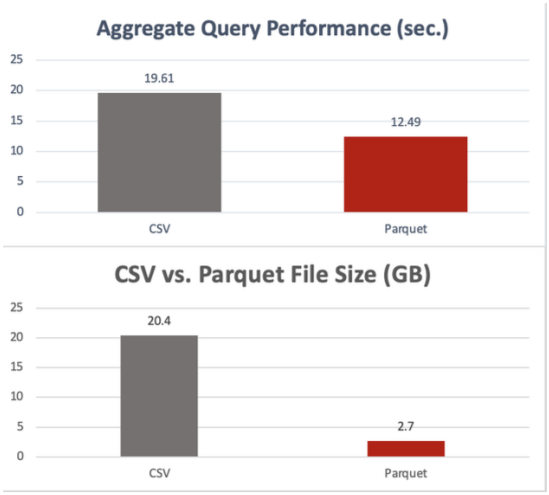


Figura 3.10 – Comparação de ingestão/análise em CSV versus Parquet [6].

O comparativo evidencia o ganho de desempenho e a redução de tamanho ao migrar de CSV para Parquet. No contexto do *Lakehouse*, esse resultado reforça a escolha por formatos colunares para consultas agregadas mais rápidas e armazenamento mais eficiente.

Em Janssen [7], a Figura 3.11 é apresentada como panorama histórico das arquiteturas de dados.

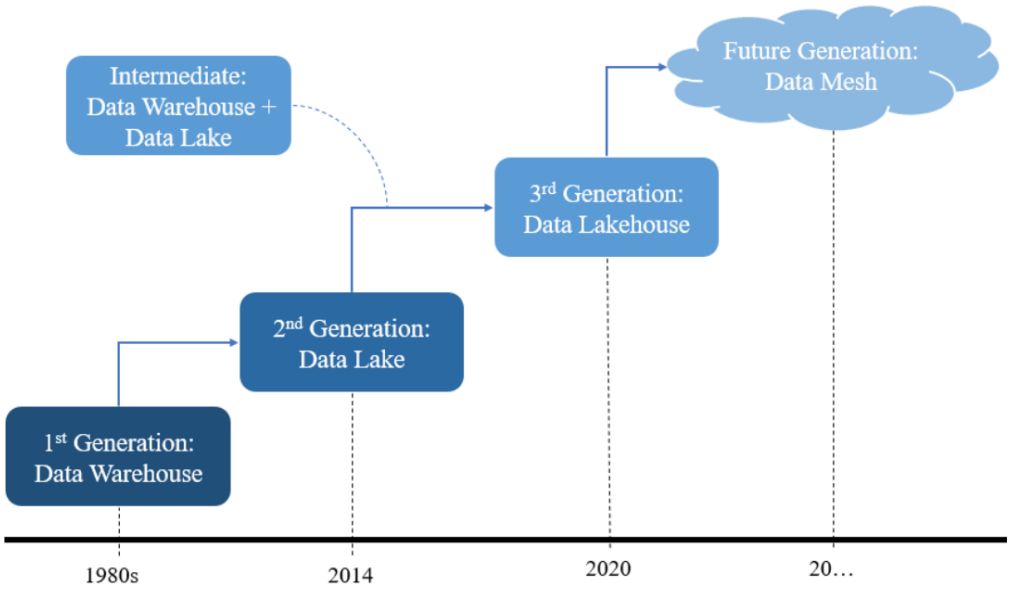


Figura 3.11 – Evolução histórica de arquiteturas de armazenamento [7].

Janssen [7] resume a evolução das arquiteturas de dados na Figura 3.11, mostrando a passagem de *data warehouse* para *data lake* e, posteriormente, para *data lakehouse*, com a perspectiva de *data mesh*. Esse encadeamento fornece o pano de fundo histórico que explica a adoção do modelo híbrido. Na sequência, a Figura 3.12 introduz a relação entre complexidade e evolução arquitetural.

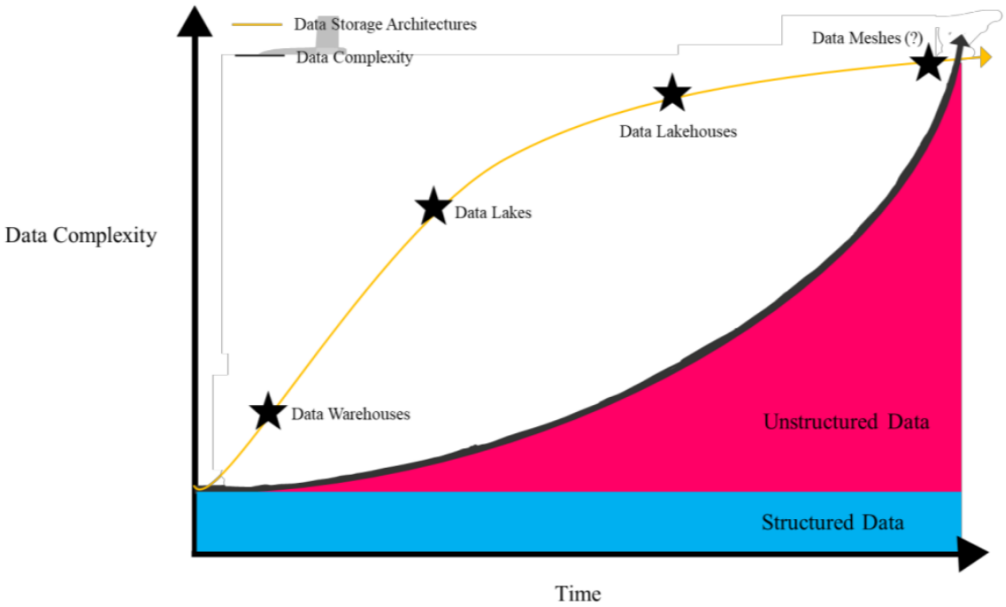


Figura 3.12 – Crescimento da complexidade e camadas de governança [7].

Complementarmente, a Figura 3.12 relaciona o aumento da complexidade dos dados

ao surgimento de novas arquiteturas, evidenciando a transição para soluções que lidam com dados não estruturados e exigências de governança mais robustas.

Ainda em Janssen [7], a Figura 3.13 é apresentada para sintetizar requisitos e limitações do *Lakehouse*.

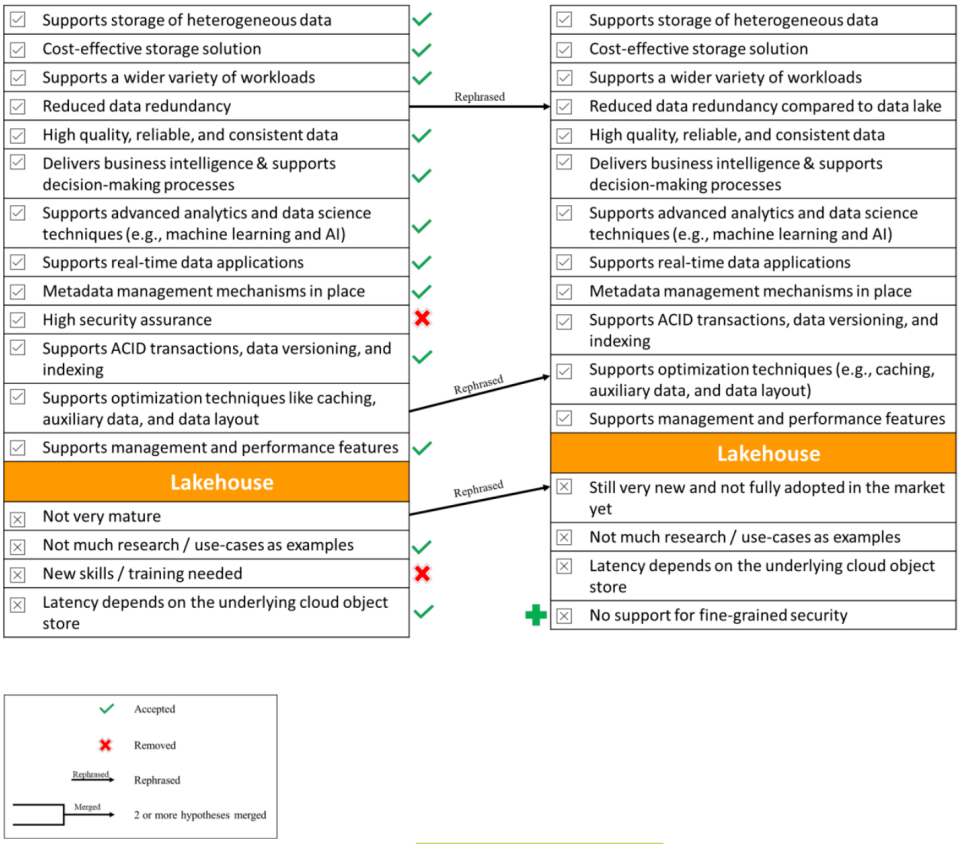


Figura 3.13 – Entidades ajustadas para unificar domínios de dados [7].

A Figura 3.13 reorganiza requisitos de *Lakehouse* em categorias, sintetizando benefícios (suporte a cargas diversas, consistência e transações) e limitações (maturidade e lacunas de adoção). A figura justifica decisões de projeto que equilibram vantagens e riscos tecnológicos. Em Rain et al. [8], a Figura 3.14 introduz a proposta de arquitetura federada.

A Figura 3.14 ilustra uma arquitetura federada com módulos de ingestão, processamento, dados confiáveis e acesso, apoiada por serviços transversais de metadados, segurança e qualidade. Esse desenho reforça a importância de um catálogo centralizado para coordenar domínios e aplicações.

Em complemento, a Figura 3.15 apresenta o detalhamento estrutural dos metadados.

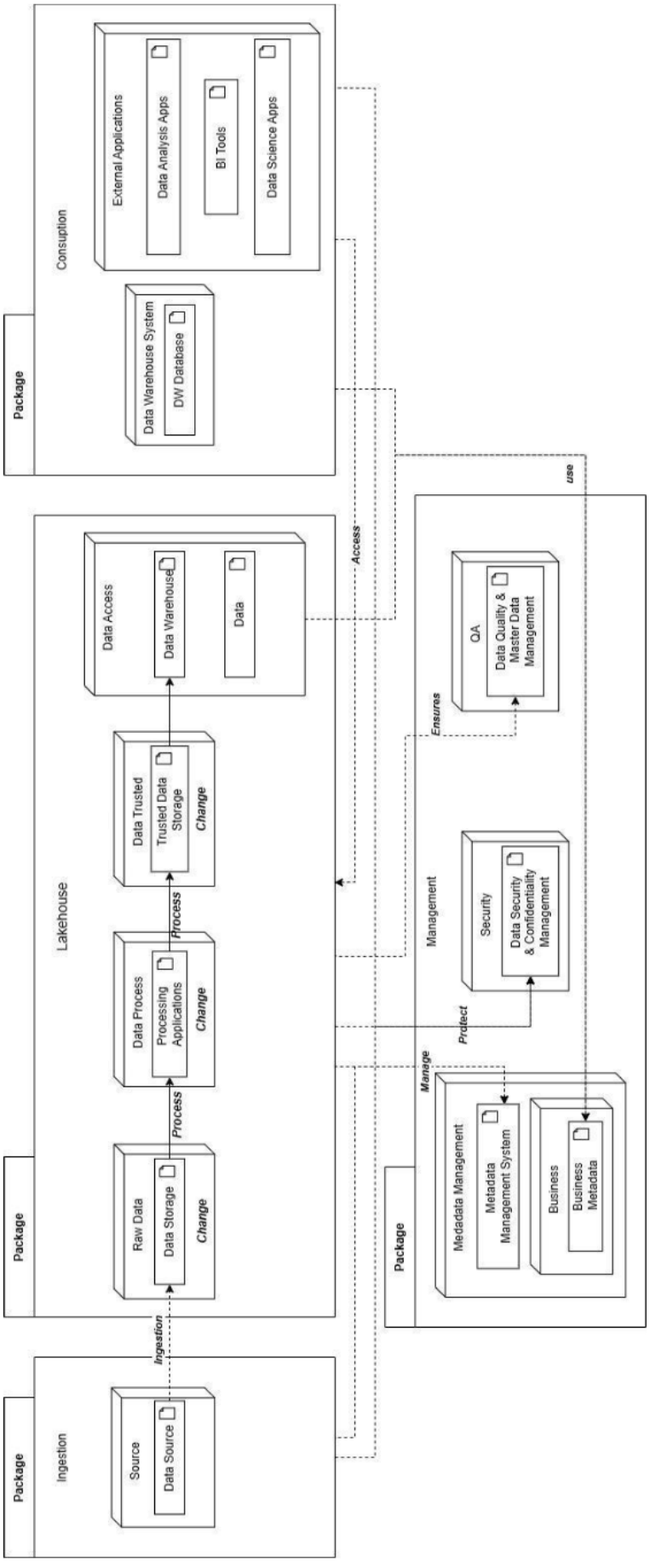


Figura 3.14 – Arquitetura federada com catálogo centralizado [8].

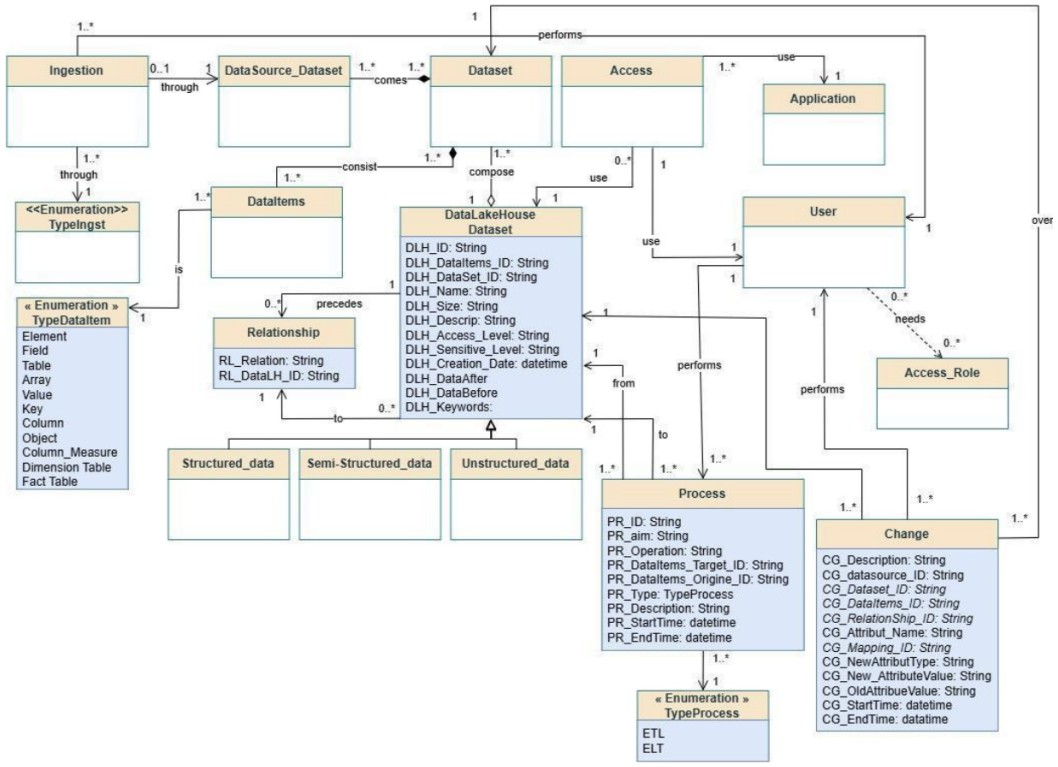


Figura 3.15 – Diagrama de classes de metadados técnicos, operacionais e semânticos [8].

O diagrama da Figura 3.15 detalha as classes de metadados e suas relações com processos, conjuntos de dados e usuários. Essa formalização sustenta a rastreabilidade e o controle de acesso em ambientes distribuídos. Em Pohl et al. [9], a Figura 3.16 introduz alternativas de modelagem para séries temporais.

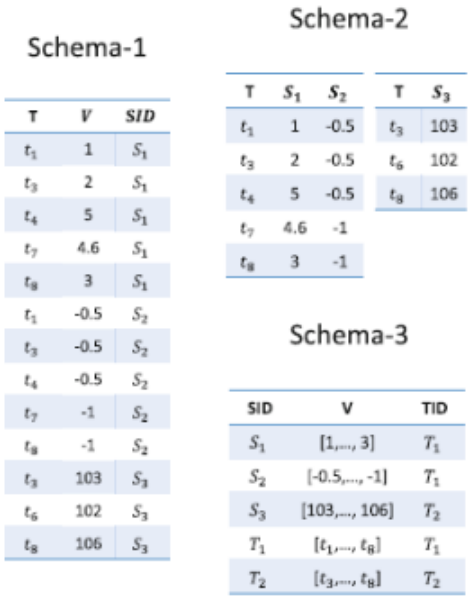


Figura 3.16 – Esquemas para séries temporais e sensores heterogêneos [9].

A Figura 3.16 compara diferentes esquemas para séries temporais, mostrando alternativas para lidar com múltiplos sensores e granularidades. A visualização auxilia na escolha de modelagens que preservem consistência e flexibilidade em alta frequência.

Na sequência, a Figura 3.17 introduz um exemplo de heterogeneidade em dados embarcados.

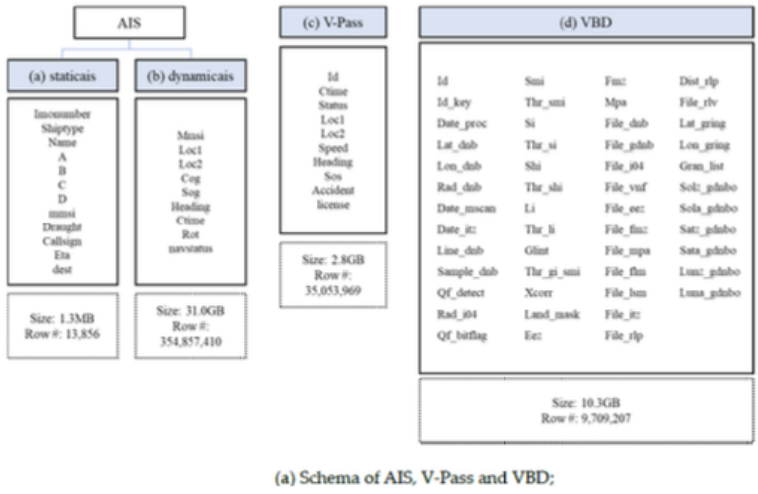


Figura 3.17 – Exemplo de dados embarcados e ingestão multimodal [9].

A Figura 3.17 exemplifica a heterogeneidade de dados embarcados (AIS, V-Pass

e VBD), com campos e tamanhos distintos. O exemplo contextualiza a necessidade de camadas de ingestão e normalização antes das análises.

Por fim, a Figura 3.18 introduz o *meta-framework* de integração de domínios.

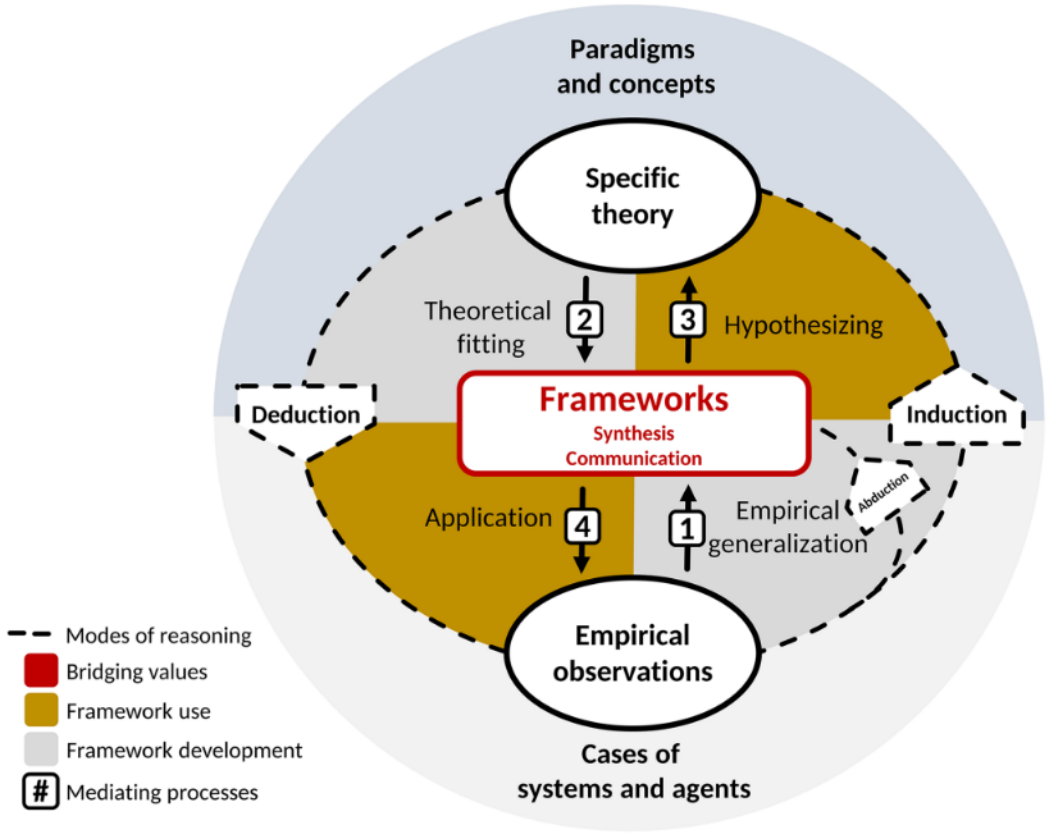


Figura 3.18 – *Meta-framework* de integração de domínios [4].

Por fim, a Figura 3.18 sintetiza um *meta-framework* que conecta observações empíricas e teoria, articulando síntese e comunicação entre domínios. O diagrama oferece base conceitual para integrar múltiplas fontes e apoiar a construção do framework proposto.

3.3 Síntese

A revisão bibliométrica identificou 35 artigos relevantes entre 2019 e 2025, com maior concentração em *SpringerLink* e *ACM* (10 cada). O *Data Lakehouse* emerge como paradigma consolidado, mas com apenas 12 menções explícitas no título, indicando maturidade conceitual ainda em evolução. Os autores analisados convergem na unificação de *Data Lakes* e *Warehouses*, com ênfase em formatos abertos (*Parquet*), transações *ACID* (*Delta Lake*, *Iceberg*) e engines unificadas (*Spark*, *Trino*). As contribuições se organizam em cinco eixos:

- **Integração** (Armbrust [5], Schneider [24]): unificação de *workloads*;
- **Governança** (Beg [6], Rain [8]): *FAIR*, qualidade e auditabilidade;
- **Organização** (Janssen [7]): camadas *bronze-silver-gold*;
- **Escalabilidade** (Kumar [47], Sun [11]): separação e compactação;
- **Séries temporais** (Pohl [9]): otimização para alta frequência.

Apesar dos avanços, há lacuna significativa na aplicação do *Data Lakehouse* a dados de sensoriamento renovável — especialmente séries temporais lacunares, heterogêneas e de alta resolução —, com pouca exploração de reconstrução inteligente, integração com IA e operação em ambientes remotos. Essa brecha justifica o desenvolvimento de uma plataforma integrada que combine *Lakehouse*, governança, camadas e *autoencoders*, como proposto neste trabalho.

4 FIDLER

Este capítulo aprofunda o desenvolvimento do FIDLER : Um *Framework* Baseado em *Data Lakehouse* para Armazenamento, Análise e Integração de Dados em Sistemas energéticos renováveis, articulando requisitos, arquitetura, metamodelos e fluxos operacionais ancorados em evidências recentes da literatura sobre *lakehouse* e aplicações energéticas [9, 10, 12, 48–51]. A ênfase recai na aplicação prática em campanhas eólicas e maremotrizes, garantindo aderência a princípios FAIR, governança e reprodutibilidade, bem como suporte a modelos de previsão e manutenção preditiva. As figuras apresentadas seguem notação UML (casos de uso, classes, atividades) em versões simplificadas para representar arquitetura e interações do framework.

4.1 Requisitos e decisões de arquitetura

O levantamento de requisitos combinou as lacunas evidenciadas no Capítulo 3 com a análise do código do projeto (backend, frontend e pipelines) e boas práticas relatadas em arquiteturas *lakehouse* para dados heterogêneos [10, 48]. A partir desse levantamento, definiram-se o escopo do sistema e os requisitos funcionais e não funcionais.

4.1.1 Requisitos (escopo do sistema)

O escopo do sistema contempla:

- Plataforma web para gestão de campanhas, equipamentos e dados de medições;
- Pipeline de ingestão e processamento para ADCP, LIDAR (ZephIR/Windcube) e SODAR;
- *Lakehouse* com camadas Bronze/Silver/Gold e integração com S3/MinIO;
- Módulos de conteúdo e comunicação (notícias, publicações, favoritos e notificações);
- Administração, auditoria e chamados de suporte.

4.1.2 Requisitos funcionais

Os requisitos funcionais, extraídos das funcionalidades implementadas nos módulos do projeto, estão sintetizados na Tabela 4.1.

Tabela 4.1 – Requisitos funcionais do FIDLER (baseados no código do projeto).

Categoria	**Requisito**	**Descrição**
Identidade e acesso	Autenticação de usuários	Cadastro, login, logout, recuperação/alteração de senha e verificação de e-mail.
Identidade e acesso	Perfil do usuário	Visualizar, atualizar, desativar e excluir conta autenticada.
Administração	Gestão de usuários	Operações de criação, leitura, atualização e exclusão (CRUD) com listagem e filtro por projeto.
Gestão de ativos	Campanhas e equipamentos	CRUD com imagem e alteração de status.
Ingestão (Bronze)	Uploads e artefatos brutos	Listar uploads/arquivos, baixar arquivo ou pacote ZIP, atualizar e excluir.
Curadoria (Silver)	Consulta e exportação de séries	Filtrar e exportar dados ADCP, SODAR e LIDAR (ZephIR/Windcube) com séries temporais.
Orquestração	Disparo de pipelines	Acionar ingestão por tipo de equipamento via API.
Processamento	Limpeza e estatísticas	Executar limpeza, normalização, *merge* e gerar estatísticas.
Produtos analíticos (Gold)	Dados sintéticos	Gerar e acompanhar status, métricas e exportações.
Conteúdo e comunicação	Publicações e notícias	CRUD e acesso por *slug* ou identificador.
Conteúdo e comunicação	Favoritos e notificações	Listar, criar, remover e marcar como lida.
Operação e auditoria	Auditoria e saúde	Consultar trilhas de auditoria e checar saúde do S3/MinIO.

4.1.3 Requisitos não funcionais

Os requisitos não funcionais priorizados e as decisões de projeto associadas são apresentados na Tabela 4.2.

Tabela 4.2 – Requisitos não funcionais priorizados para o FIDLER e decisões de projeto.

Categoria	**Requisito**	**Decisão de projeto**
Interoperabilidade	Ingestão de CSV, NetCDF, JSON e streaming	*Schema-on-read* na Bronze com conversão para Parquet/Delta [10]
Governança	Rastreamento de versão e *lineage*	Delta Lake + *time travel* e catálogo interno com trilha de auditoria [48]
Latência	Consultas sub-segundo em painéis	Particionamento temporal e *caching* na Gold [49]
IA/ML	Suporte a previsão e detecção de anomalias	Pipelines PySpark + Prophet/XGBoost no Bronze-Silver-Gold [9, 12]
Resiliência	Operação com conectividade intermitente	Ingestão assíncrona com *staging* local e replicação eventual [50]
Segurança	Controle de acesso federado	RBAC com JWT/OAuth2 e trilhas de auditoria [51]

4.2 Arquitetura em camadas do FIDLER

A arquitetura adota o padrão *lakehouse* em três zonas (Bronze, Silver e Gold), combinando *schema-on-read* para flexibilidade e *schema-on-write* para consumo analítico [10]. Os diagramas desta seção foram elaborados pelo autor com base nos princípios sintetizados no Capítulo 3. A Figura 4.1 apresenta a visão geral dessa organização em camadas.

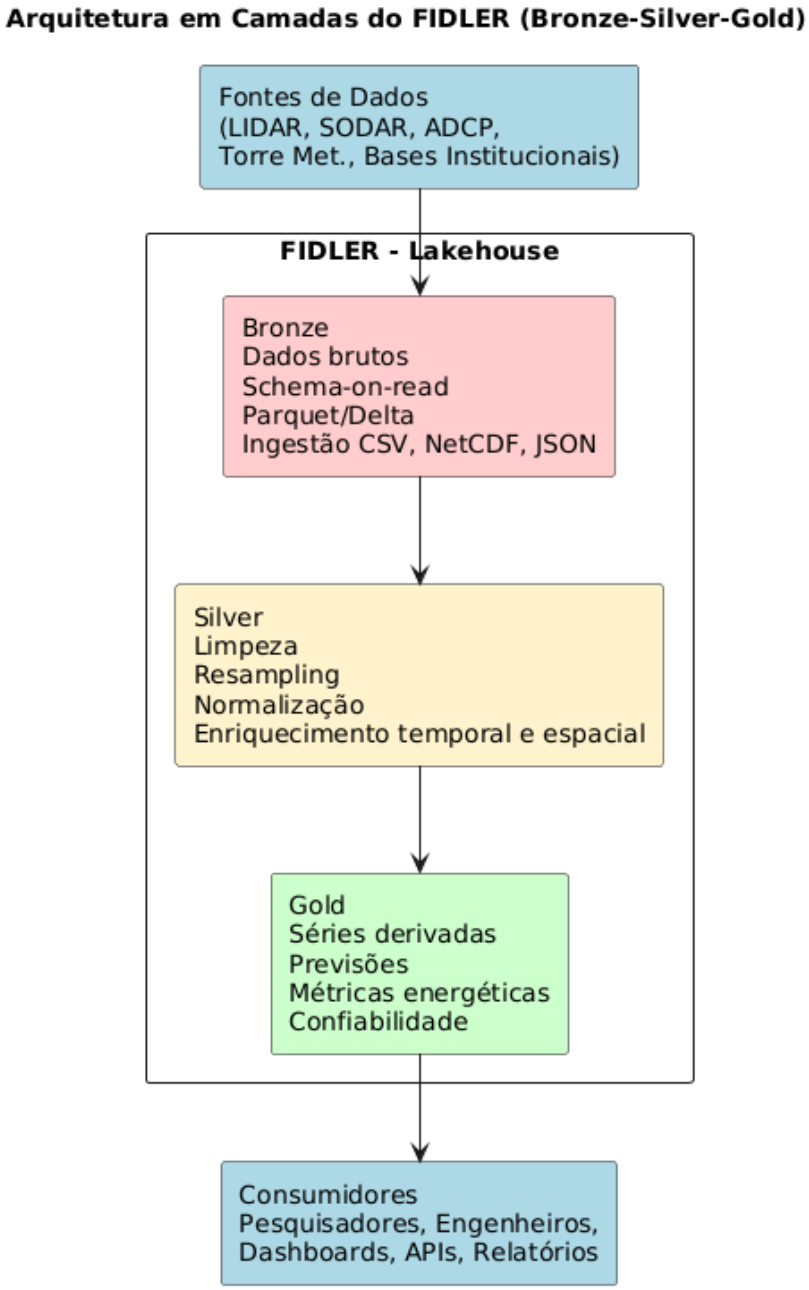


Figura 4.1 – Visão geral da arquitetura FIDLER com zonas Bronze, Silver e Gold.

Observa-se a cadeia completa de valor, desde as fontes de dados (sensores e bases institucionais), passando pelas zonas Bronze, Silver e Gold, até os consumidores finais, o que reforça a separação entre dados brutos, dados curados e produtos analíticos.

A Figura 4.2 detalha a Camada Bronze, explicitando a *landing zone* e o registro de metadados associados à ingestão.

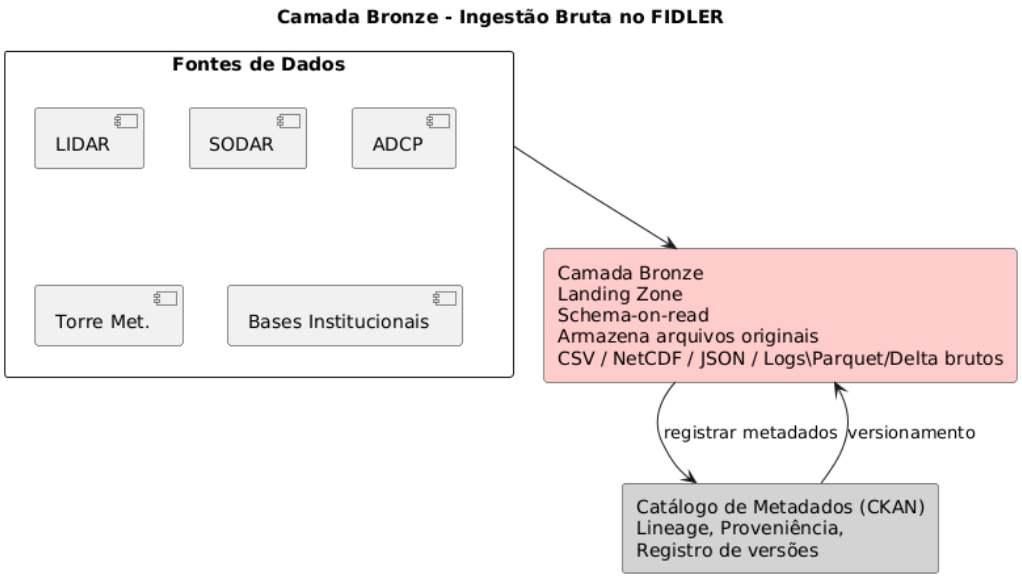


Figura 4.2 – Camada Bronze: ingestão bruta de LIDAR, SODAR, ADCP e bases institucionais.

Nesse estágio, os arquivos originais (CSV, NetCDF, JSON e logs) são preservados com *schema-on-read*, enquanto o catálogo registra *lineage*, proveniência e versões para garantir rastreabilidade.

A sequência das figuras de ingestão e transformação reforça que o catálogo captura metadados detalhados (sensor/campanha, arquivo, coordenadas, timestamp, hash, operador, status de qualidade e versão de pipeline). Esses atributos são usados na Silver para aplicar regras de consistência e na Gold para atestar que indicadores (REWS, curvas de potência, estimativas de geração) estão vinculados à origem física das medições, tornando possível explicar auditorialmente cada valor entregue aos investidores.

A Figura 4.3 apresenta a Camada Silver, dedicada à padronização, validação e curadoria das séries temporais.

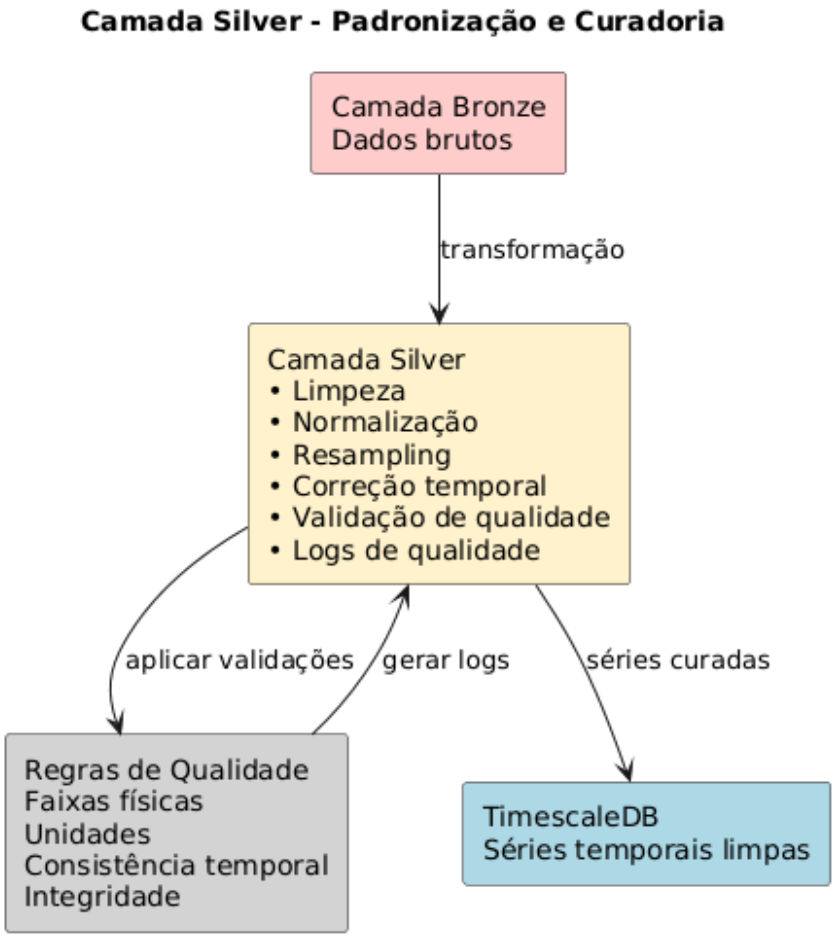


Figura 4.3 – Camada Silver: padronização, limpeza, enriquecimento temporal e espacial.

O diagrama evidencia a aplicação de regras de qualidade (faixas físicas, unidades e consistência temporal), bem como a geração de séries limpas em banco temporal, assegurando consistência para análises posteriores.

A Figura 4.4 descreve a Camada Gold, onde são produzidos indicadores, previsões e métricas energéticas.

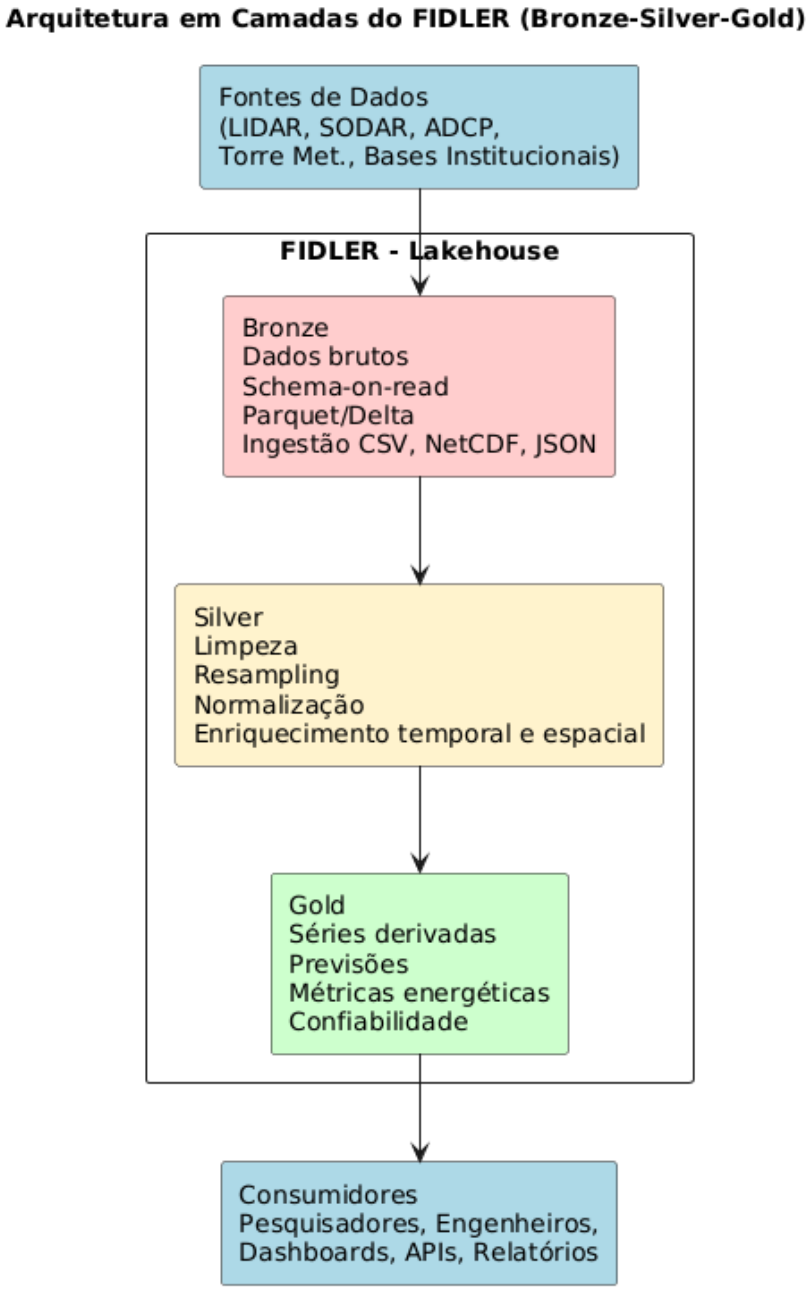


Figura 4.4 – Camada Gold: séries derivadas, previsões e métricas de confiabilidade.

Os artefatos dessa camada consolidam séries derivadas e produtos analíticos, permitindo o consumo por painéis, APIs e relatórios com foco decisório.

O fluxo de ingestão é introduzido na Figura 4.5, que organiza a entrada de dados a partir de múltiplas fontes e a conexão com o catálogo.

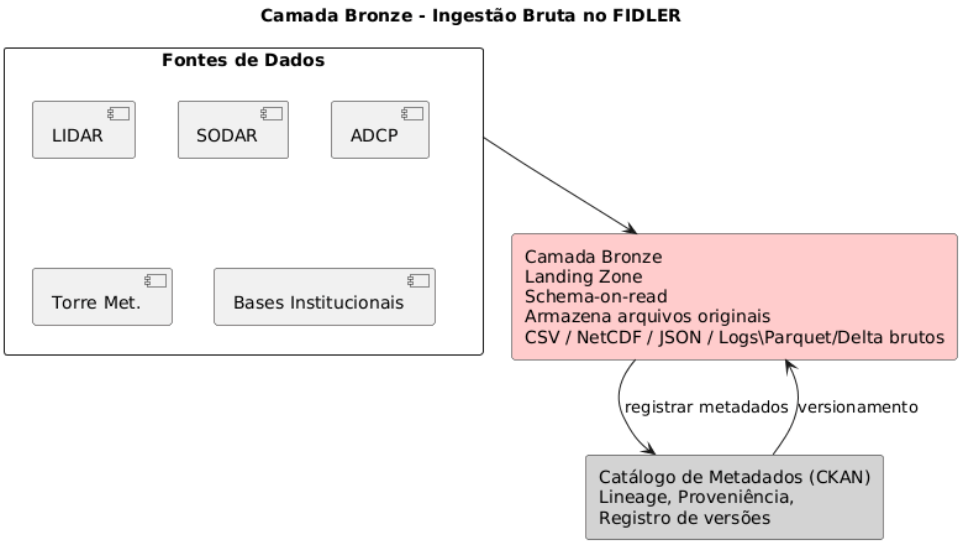


Figura 4.5 – Fluxo de ingestão parametrizável em modo lote e *streaming*.

O esquema destaca o registro automático de metadados e versionamento, assegurando trilha de auditoria desde a camada de entrada.

A Figura 4.6 apresenta o fluxo de transformação da Silver, com etapas de limpeza e verificação de qualidade.

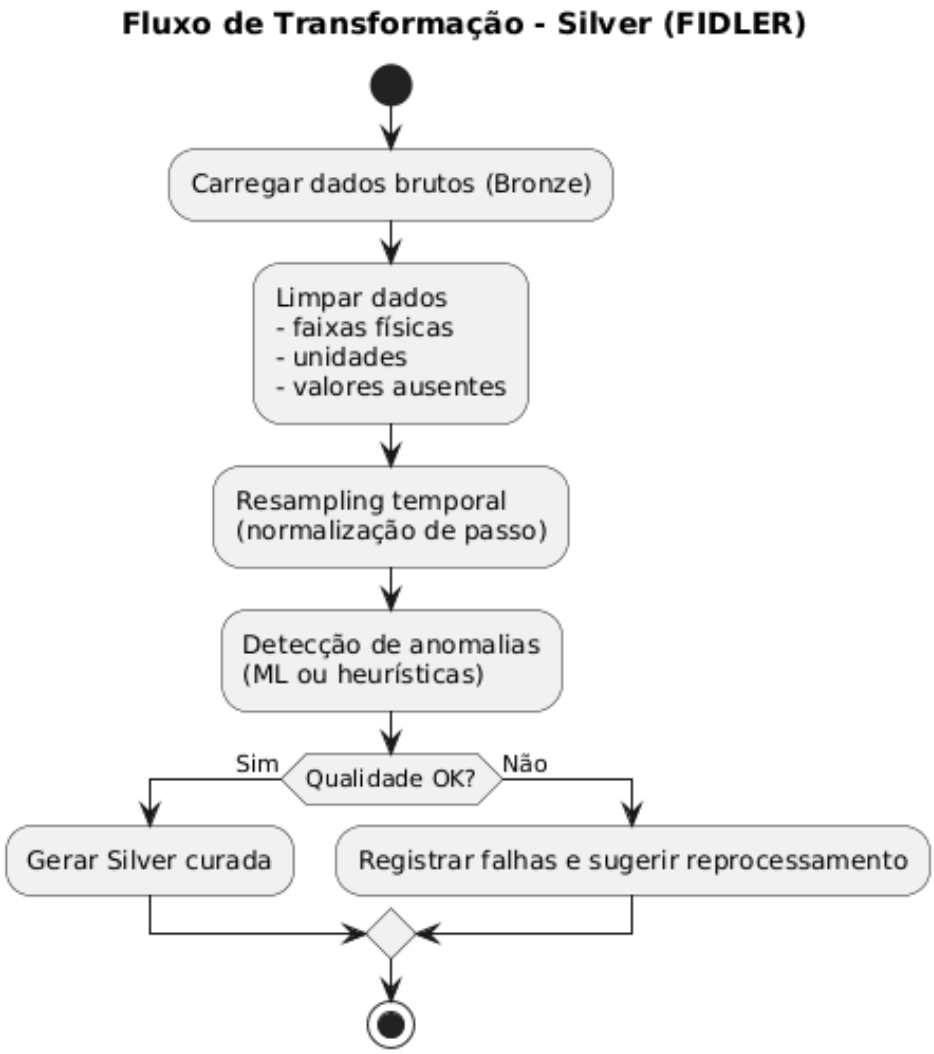


Figura 4.6 – Fluxo de transformação: limpeza, *resampling*, detecção de anomalias e curadoria.

O diagrama explicita o encadeamento de limpeza, *resampling*, detecção de anomalias e geração de logs de qualidade antes da promoção para séries curadas.

A Figura 4.7 sintetiza o fluxo de consumo, destacando os canais de exposição dos produtos da Gold.

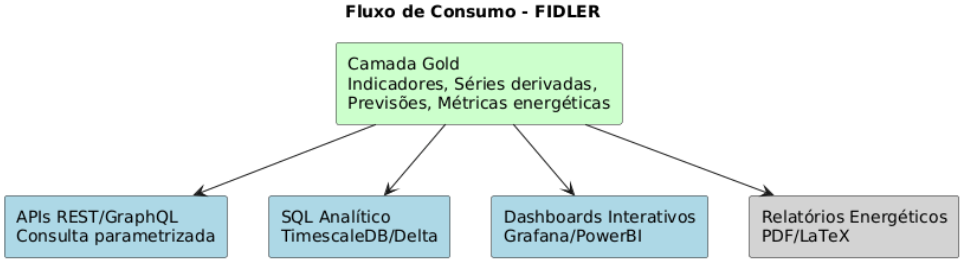


Figura 4.7 – Fluxo de consumo: APIs, SQL analítico e painéis interativos.

Nesse estágio, os indicadores e séries derivadas são disponibilizados via APIs REST/GraphQL, SQL analítico e *dashboards*, atendendo perfis técnicos e decisores.

A Figura 4.8 introduz o diagrama de casos de uso do framework prospectivo, elaborado pelo autor a partir dos requisitos deste capítulo.

São identificados os seguintes atores:

- **Técnico de Campo**: responsável por coletar e anotar dados diretamente no sistema;
- **Pesquisador**: realiza curadoria, harmonização, análise e visualização dos dados;
- **Administrador**: responsável por permissões, catálogo e auditorias;
- **Consumidor Público**: dedicado à consulta de dados abertos.

Os casos de uso são organizados por camada:

- **Camada Bronze:**
  - Ingerir dados brutos;
  - Anotar eventos temporais.
- **Camada Prata:**
  - Curar e harmonizar dados;
  - Gerar séries derivadas.
- **Camada Ouro:**
  - Gerar produtos analíticos;
  - Visualizar *dashboards*;
  - Executar modelo preditivo;
  - Consultar séries temporais;
  - Consumir APIs/dados.
- **Governança Federada:**
  - Gerenciar permissões;
  - Gerenciar catálogo de metadados;
  - Auditar logs;
  - Atribuir identificadores persistentes.
- **Consulta Pública:**

– Consultar dados públicos.

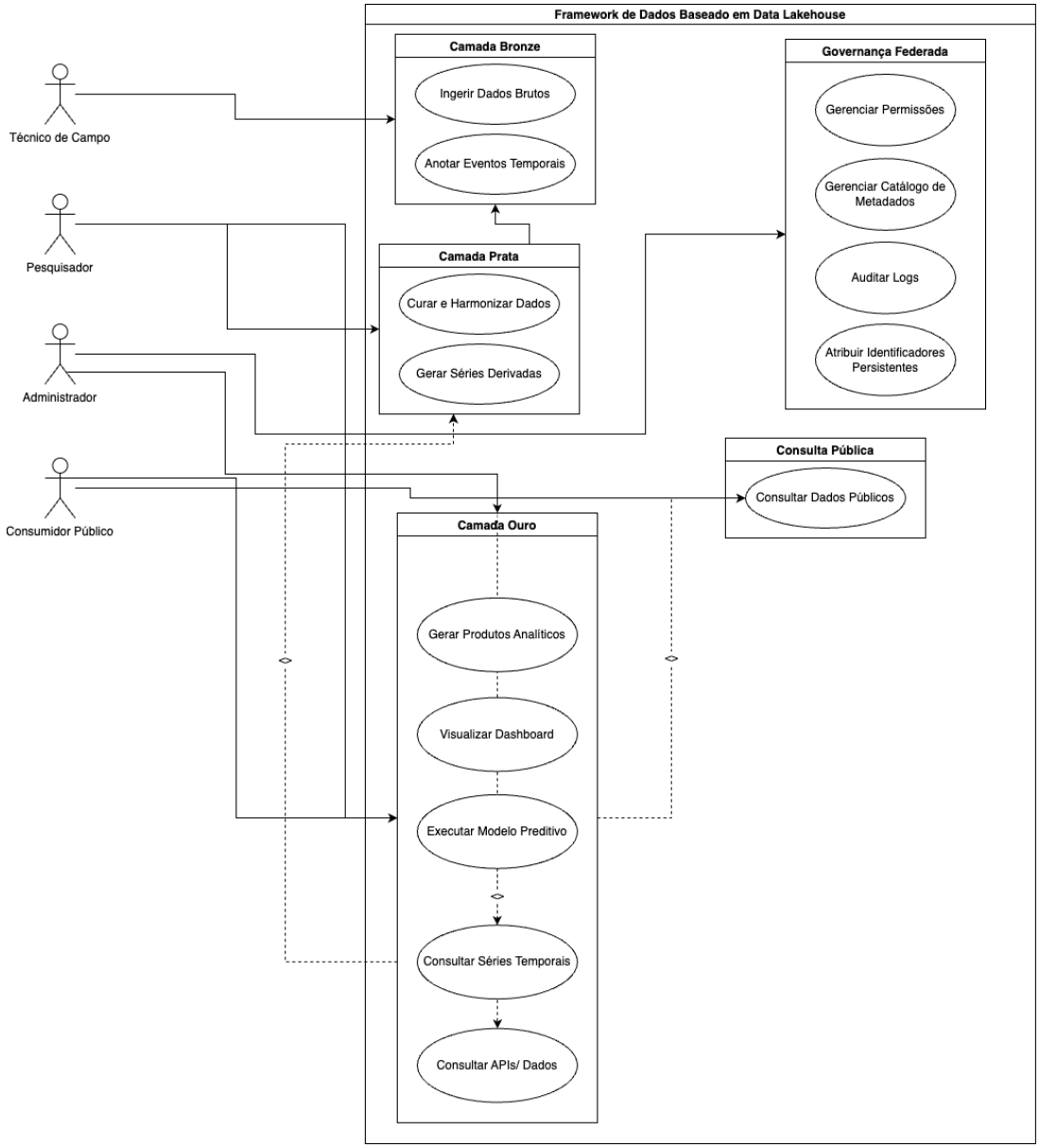


Figura 4.8 – Diagrama de casos de uso do FIDLER com atores e fluxos principais.

O diagrama evidencia as fronteiras funcionais entre camadas operacionais e de governança, reforçando o papel central do administrador na conformidade e da camada Gold na entrega de produtos analíticos.

4.3 Metamodelos e camadas lógicas

Os metamodelos do FIDLER foram definidos para garantir reutilização e rastreabilidade, e os diagramas desta seção foram elaborados pelo autor com base

nas necessidades de governança e séries temporais. A Tabela 4.3 resume as principais entidades e artefatos associados às zonas.

Tabela 4.3 – Metamodelos e artefatos por zona do FIDLER.

Zona	**Entidades principais**	**Artefatos e padrões**
Bronze	Sensor, Dataset Bruto, Metadado Técnico	Parquet/Delta brutos; schema flexível; *landing* assíncrono
Silver	Série Temporal, Transformação, Log de Qualidade	Tabelas particionadas; logs de parâmetros; regras de consistência [9]
Gold	Série Derivada, Modelo, Previsão, Evento	Tabelas dimensionais; métricas de previsão; anotações de manutenção

A Figura 4.9 apresenta o diagrama de classes simplificado que relaciona sensores, datasets, transformações e previsões, elaborado pelo autor com base nas necessidades de rastreabilidade do FIDLER.

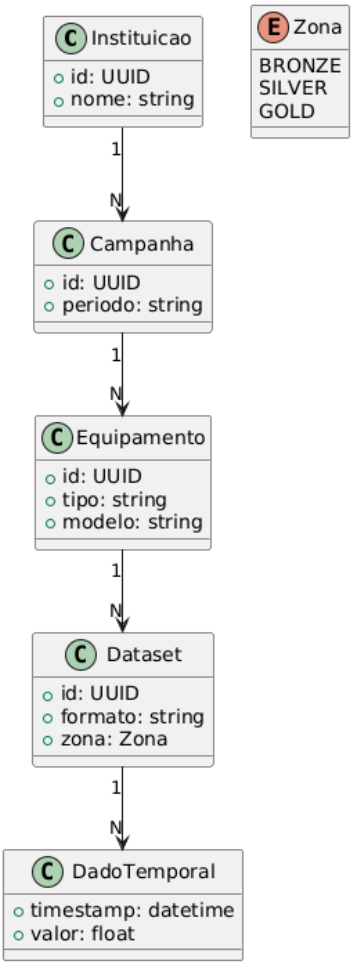


Figura 4.9 – Metamodelo de classes: sensores, datasets, transformações e previsões.

O diagrama explicita a ligação entre a camada de ingestão (sensores e pipeline), armazenamento em zonas Bronze/Prata, governança (catálogo e instituições) e processamento/consulta, reforçando a separação entre dados, controle de acesso e exposição.

A Figura 4.10 introduz o fluxo de atividades com *swimlanes* por papel, alinhado ao padrão de automação incremental.

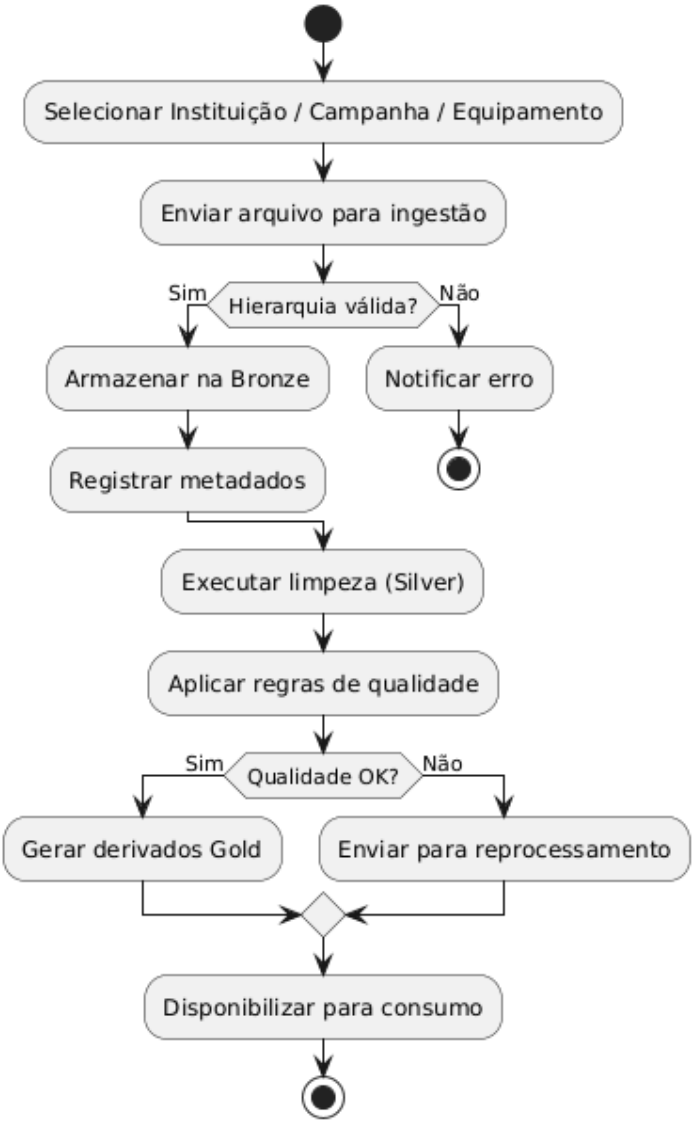


Figura 4.10 – Metamodelo de atividades com *swimlanes* por papel.

O fluxo descreve a validação da hierarquia institucional, a persistência na Bronze, a limpeza na Silver, a aplicação de regras de qualidade e a promoção para Gold, com desvios para reprocessamento quando necessário.

4.4 Componentização e tecnologias

O FIDLER adota componentes substituíveis, priorizando tecnologias abertas para reduzir *vendor lock-in*. A Tabela 4.4 mapeia funções a ferramentas.

Tabela 4.4 – Componentes tecnológicos adotados no FIDLER.

Função	**Tecnologia**	**Justificativa**
Ingestão	Spark Structured Streaming / Auto Loader	Robustez para múltiplos formatos e *back-pressure* [10]
Armazenamento bruto	Delta Lake em armazenamento de objetos	ACID + *time travel* e custo eficiente [48]
Banco relacional (opcional)	PostgreSQL	Consultas rápidas em séries e suporte a relatórios operacionais [49]
Catálogo e auditoria	Módulo interno + ontologias energéticas	FAIR, rastreabilidade e controle de mudanças [51]
Orquestração	Jobs agendados (quando necessário)	Pipelines controlados e repetíveis [9]
Observabilidade	Prometheus + Grafana	Métricas e rastreabilidade operacionais [50]
APIs	FastAPI (REST) + Swagger	Baixa latência e documentação integrada para consumo

A avaliação de armazenamento compara Amazon S3 (dados brutos/parquet) e PostgreSQL (opcional, para curadoria e consultas relacionais). A Tabela 4.5 resume critérios e a motivação para manter ambos durante o período de testes antes da migração definitiva.

Tabela 4.5 – Comparativo prático S3 vs. PostgreSQL (opcional) no FIDLER.

Critério	**S3 (objeto)**	**PostgreSQL**	**Observação**
Custo/GB	Baixo, elástico	Moderado, ligado a IOPS	S3 para histórico bruto; PG para curadoria
Latência consulta	Alta (scan)	Baixa (índices)	Painéis sensíveis usam PostgreSQL (quando disponível)
Versionamento	Delta Lake/*time travel*	Temporal nativo limitado	Delta cobre rollback de bronze/silver
Formato	Parquet/Delta	Tabelas relacionais	Convivência facilita comparação e migração futura
Integração	Nativa com Spark	Nativa com BI SQL	Pipelines leem ambos para A/B de desempenho

A Figura 4.11 introduz a implantação de referência em nuvem ou VM local, com serviços conteinerizados e isolamento básico.

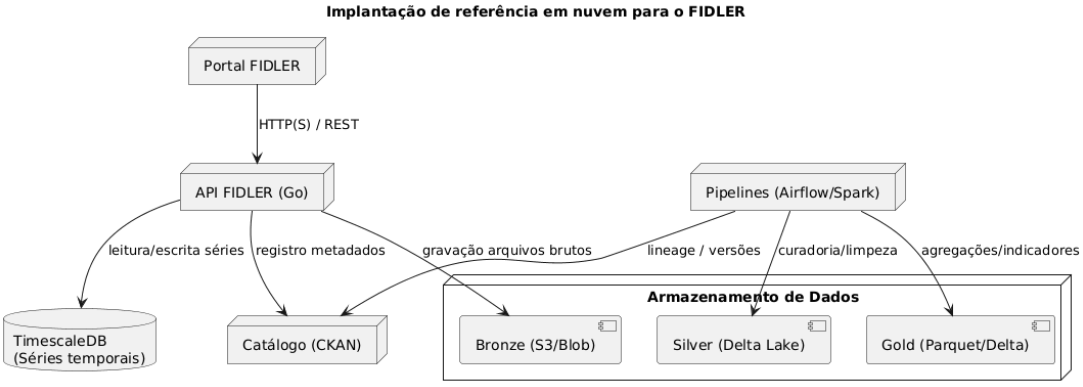


Figura 4.11 – Implantação de referência em nuvem ou VM local com serviços conteinerizados.

O desenho evidencia a separação entre portal, API, pipelines e armazenamento em zonas, além da integração com catálogo de metadados e banco temporal para consultas especializadas.

4.5 Qualidade de dados, governança e segurança

As regras de qualidade de dados são aplicadas incrementalmente na Silver. A Tabela 4.6 lista checagens principais e ações corretivas.

Tabela 4.6 – Regras de qualidade e ações automatizadas na camada Silver.

Regra	**Verificação**	**Ação**
Disponibilidade	% de completude por altura/variável	Marcar lacunas, gerar série sintética ou sinalizar reprocessamento
Consistência	Unidades e faixas físicas válidas	Normalizar unidades, descartar pontos fora da física
Temporal	Monotonicidade e *time drift*	Ajustar fuso/latência, reindexar
Espacial	Coordenadas dentro do polígono do parque	Corrigir metadados ou rejeitar lote

A Tabela 4.7 resume mecanismos de governança e segurança alinhados a [51].

Tabela 4.7 – Mecanismos de governança e segurança implementados.

Mecanismo	**Camada**	**Descrição**
RBAC com JWT/OAuth2	API e catálogo	Perfis de pesquisador, operador e investidor
Encriptação	Armazenamento/Trânsito	Chaves gerenciadas, TLS e criptografia em repouso
Auditoria	Todas	Logs imutáveis via Delta *change feed*
Mascaramento	Gold	Pseudonimização de dados sensíveis de operação

4.6 IA/ML e automação

A camada de IA foi concebida para dois objetivos principais: reconstrução de dados faltantes e previsão de valores futuros. Para a reconstrução, foram adotadas abordagens baseadas em autoencoders, com referência a arquiteturas convolucionais para reconstrução e remoção de ruído [43], métodos adversariais condicionais para reconstrução de sinais [44] e técnicas de redução de dimensionalidade com autoencoders [45]. Essas abordagens aprendem padrões latentes da série e permitem restaurar trechos ausentes com boa consistência; quanto maior e mais representativa for a base histórica, melhor tende a ser a reconstrução, pois o modelo aprende um repertório mais amplo de comportamento.

Esse mecanismo é essencial para manter a continuidade do histórico sem perder coerência física. Já para previsão, adotou-se uma estratégia de geração de horizonte temporal configurável, com janelas específicas de 1 dia, 1 semana e 1 mês. A lógica parte do último trecho confiável da série e extrapola a tendência temporal a curto e médio prazo, garantindo previsões estáveis e interpretáveis. Em geral, com a mesma base de treino, previsões de curto prazo (1 dia) apresentam maior precisão do que previsões de 1 semana ou 1 mês, pois a incerteza cresce com o horizonte temporal. Esse desenho segue a literatura de forecasting com redes recorrentes (LSTM/GRU) e técnicas baseadas em janelas deslizantes, onde cada horizonte (dia/semana/mês) é tratado como um problema de projeção distinto, permitindo calibração de granularidade e controle de erro.

Assim, a IA não apenas corrige lacunas históricas, como também entrega projeções úteis para decisões operacionais, com previsões de curto prazo (1 dia) para acompanhamento imediato, médio prazo (1 semana) para planejamento operacional e longo prazo (1 mês) para análises de tendência e tomada de decisão estratégica.

Os pipelines de IA seguem níveis de automação propostos em [12], cobrindo desde ingestão até monitoramento de modelos. A Figura 4.12 apresenta o fluxo de MLOps simplificado para previsões de geração e manutenção preditiva.

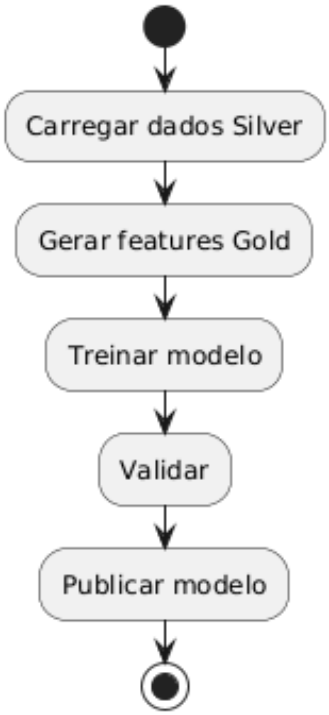


Figura 4.12 – Fluxo MLOps para previsão de geração e manutenção preditiva.

O diagrama sintetiza o carregamento de dados Silver, a geração de *features* na Gold, o treinamento, a validação e a publicação de modelos.

A Tabela 4.8 resume modelos recomendados e casos de uso.

Tabela 4.8 – Modelos e casos de uso suportados no FIDLER.

Modelo	**Caso de uso**	**Configuração recomendada**
Prophet	Previsão horária/dia-ahead de geração	Tendência + sazonalidades diurna/semana; regressoras meteorológicas
XGBoost	Manutenção preditiva (anomalías)	Janelas deslizantes de vibração/temperatura; *early stopping*
LSTM	Séries multivariadas de vento/maré	48–96 *lags*, normalização min-max, *dropout* 0,2

4.7 Escalabilidade e resiliência

Tomando como referência [50] para armazenamento de alto desempenho, o FIDLER adota replicação eventual e *auto-scaling* moderado para equilibrar custo e disponibilidade. A Tabela 4.9 apresenta metas de SLO/SLA e mecanismos de resiliência.

Tabela 4.9 – Metas de SLO/SLA e mecanismos de resiliência.

Indicador	**Meta**	**Mecanismo**
Disponibilidade dados Gold	99,5% mensal	Replicação entre zonas + *time travel* Delta
Latência consulta painéis	$<$ 800 ms p95	Particionamento temporal + *caching*
RPO ingestão Bronze	1 hora	*Staging* local + *checkpoint* Spark
RTO Silver $\rightarrow$ Gold	15 min	Pipeline incremental agendado

4.8 Diagramas UML alinhados aos marcos do projeto

Esta seção consolida os principais diagramas UML do FIDLER, conectando de forma explícita os requisitos não funcionais da Tabela 4.2, a arquitetura em camadas apresentada nas Figuras 4.1–4.7, os metamodelos de dados da Tabela 4.3 e os componentes tecnológicos da Tabela 4.4. Todos os diagramas foram elaborados pelo autor no padrão UML, a partir das estruturas conceituais de *Data Lakehouse* discutidas no Capítulo 3 [5–9, 11, 24].

4.8.1 Arquitetura conceitual UML do FIDLER

A Figura 4.13 apresenta a arquitetura conceitual do FIDLER em notação UML de pacotes. À esquerda, o pacote *Fontes de Dados* agrupa sensores de campo (LIDAR, SODAR, ADCP, torres micrometeorológicas) e bases institucionais (ONS, INMET, centros de pesquisa), representando a origem física e institucional das séries temporais de vento e maré. No centro, o pacote *FIDLER – Núcleo Lakehouse* encapsula as zonas Bronze, Silver e Gold e os módulos de Ingestão, Transformação, Governança e IA/ML, refletindo as decisões de projeto da Tabela 4.2. À direita, o pacote *Consumidores* representa perfis de uso (pesquisador, planejador, regulador, investidor) que consomem os produtos analíticos gerados.

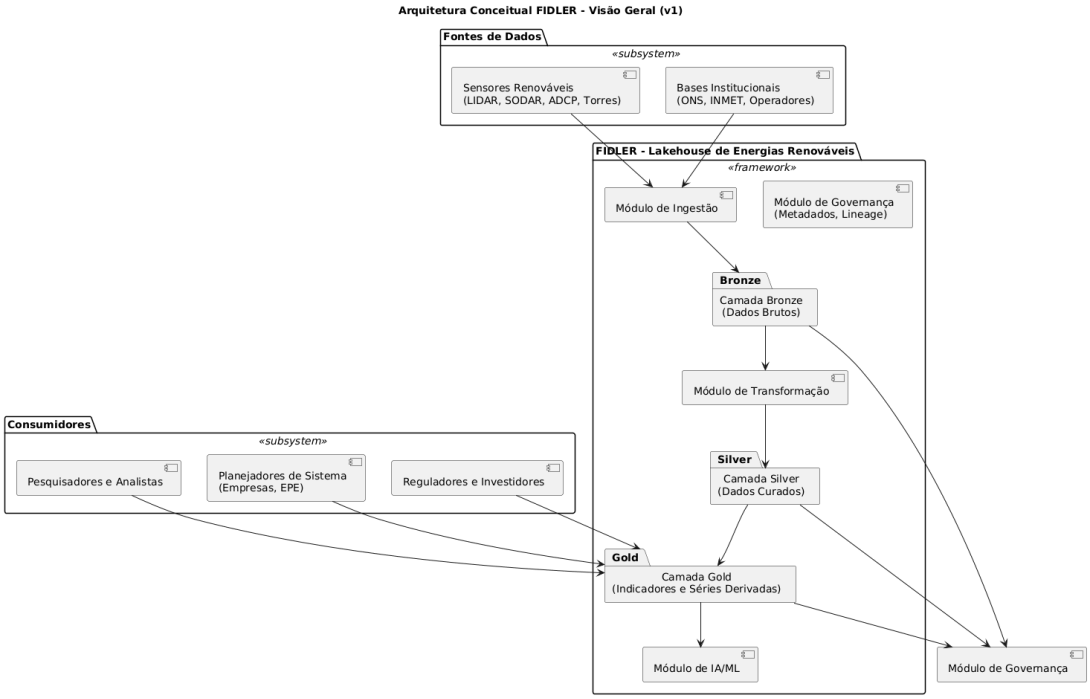


Figura 4.13 – Arquitetura conceitual UML do FIDLER, em termos de pacotes: fontes de dados (sensores e bases institucionais), núcleo *lakehouse* (Bronze, Silver, Gold, governança, IA/ML) e consumidores. Elaborado pelo autor a partir das arquiteturas de *Data Lakehouse* discutidas em [5, 7].

As dependências entre pacotes deixam claro que todos os fluxos começam em instituições que operam campanhas e equipamentos, passam obrigatoriamente pelo núcleo *lakehouse* do FIDLER e só então alcançam os consumidores. Este diagrama sintetiza, em um único quadro, a visão de contexto de alto nível do sistema.

4.8.2 Diagrama de casos de uso (visão funcional)

A Figura 4.14 mostra o diagrama de casos de uso do FIDLER. O sistema FIDLER é representado como uma fronteira que encapsula as funcionalidades principais, interagindo com os seguintes atores externos:

- ***Administrador FIDLER***: configura instituições, campanhas e permissões;
- ***Operador de Campo***: solicita ingestão de dados de campanhas;
- ***Engenheiro de Dados***: configura *pipelines* e monitora qualidade dos dados;
- ***Cientista de Dados***: treina modelos e executa análises;
- ***Pesquisador/Analista***: consulta séries temporais e gera relatórios.

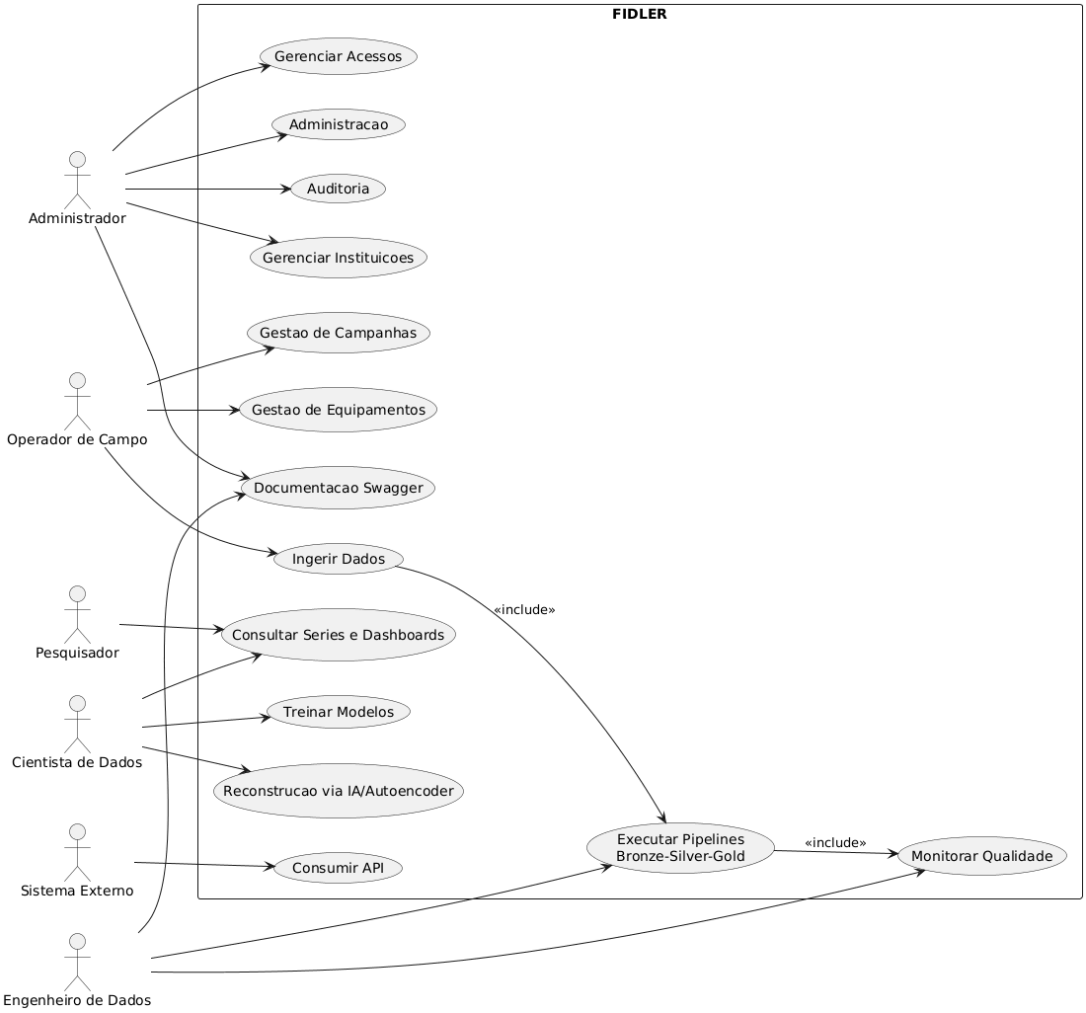


Figura 4.14 – Diagrama de casos de uso do FIDLER, com atores (administrador, operador de campo, engenheiro de dados, cientista de dados, pesquisador) e casos de uso centrais (gestão de domínio, ingestão, pipelines Bronze–Silver–Gold, IA/ML, consulta e governança). Elaborado pelo autor com base nos requisitos funcionais e de governança discutidos no capítulo.

Os casos de uso do sistema implementam três dimensões funcionais principais:

- **Dimensão Hierárquica e Administrativa:**
  - "Gerenciar Instituições", "Gestão de Campanhas" e "Gestão de Equipamentos" materializam a hierarquia obrigatória *Instituição* → *Campanha* → *Equipamento*.
- **Dimensão Operacional e de Processamento:**
  - "Ingerir Dados", "Executar Pipelines Bronze–Silver–Gold" e "Monitorar Qualidade" representam o fluxo operacional de ingestão e curadoria.
- **Dimensão Analítica e de Acesso:**

– “Treinar Modelos”, “Reconstrução via IA/Autoencoder”, “Consultar Séries e Dashboards”, “Consumir API” e “Gerenciar Acessos” conectam o sistema aos requisitos de IA/ML, baixa latência e segurança apresentados na Tabela 4.2.

4.8.3 Diagrama de classes

A Figura 4.15 apresenta o diagrama de classes que formaliza o metamodelo de dados do FIDLER. A classe `Instituicao` relaciona-se com a classe `Campanha` por uma associação 1:N, indicando que uma instituição pode conduzir diversas campanhas, mas nenhuma campanha existe sem instituição associada. A classe `Campanha`, por sua vez, relaciona-se com `Equipamento` também em 1:N, assegurando que nenhum equipamento exista fora de uma campanha.

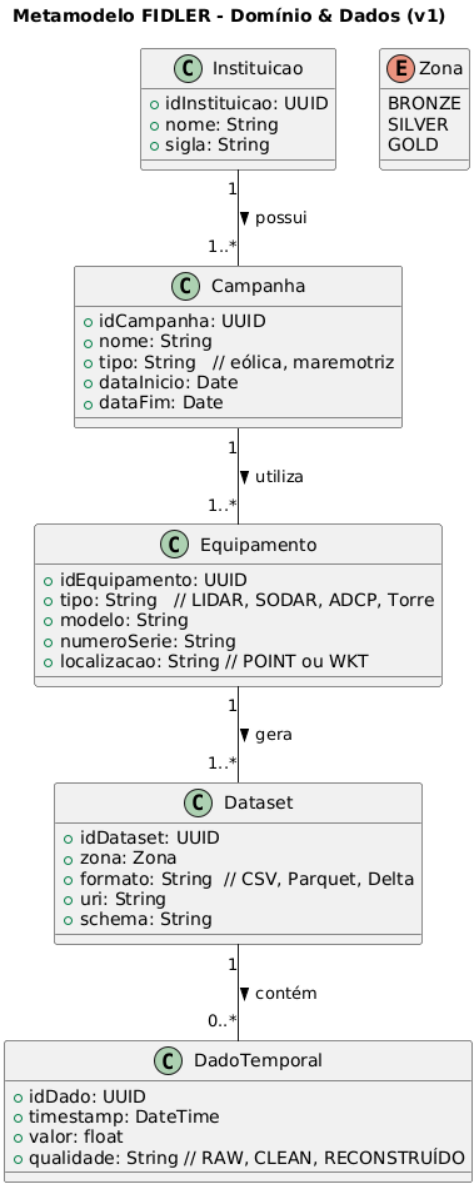


Figura 4.15 – Diagrama de classes do FIDLER, formalizando a hierarquia Instituição → Campanha → Equipamento → Dataset → DadoTemporal, com enumeração de zonas Bronze–Silver–Gold. Elaborado pelo autor com base nas práticas de modelagem para séries temporais em ambientes *lakehouse* [9].

Cada `Equipamento` se associa a um ou mais `Dataset`, que representam conjuntos de arquivos ou tabelas vinculados ao equipamento. A classe `Dataset` possui um atributo `zona` do tipo enumerado `Zona`, com valores `BRONZE`, `SILVER` e `GOLD`, alinhando-se diretamente à arquitetura em camadas da Seção 2. Por fim, a classe `DadoTemporal` representa os registros individuais de séries temporais, associados obrigatoriamente a um `Dataset`. As multiplicidades explicitam a regra de negócio essencial: não há dado sem dataset, nem dataset sem equipamento, nem equipamento sem campanha e nem campanha sem instituição.

4.8.4 Diagrama de atividades (ingestão, transformação e consumo)

A Figura 4.16 apresenta o diagrama de atividades que descreve o fluxo de ingestão, transformação e consumo de dados no FIDLER. O fluxo inicia com o recebimento de dados de campanha pela API, seguido da validação da hierarquia Instituição–Campanha–Equipamento; em caso de inconsistência, o registro é marcado como rejeitado.

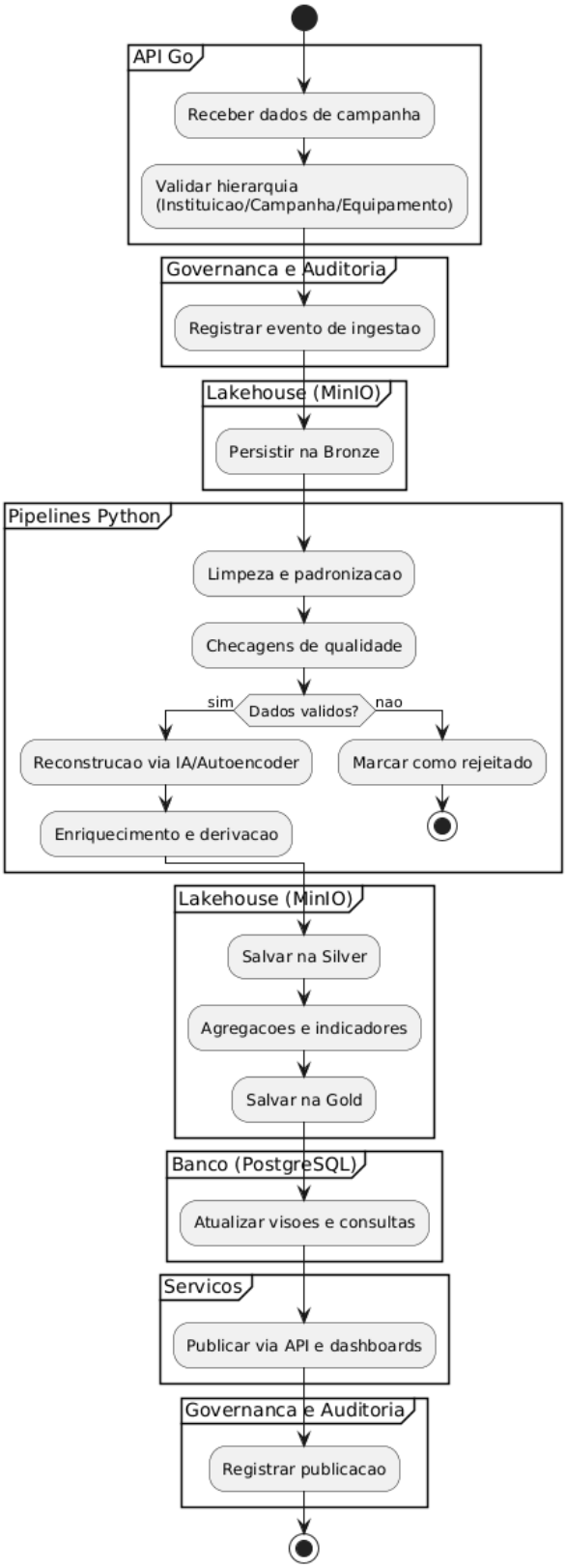


Figura 4.16 – Diagrama de atividades do FIDLER, detalhando o fluxo de ingestão (validação de hierarquia), promoção Bronze → Silver → Gold, checagens de qualidade e disponibilização para consumo. Elaborado pelo autor a partir das estratégias de pipelines em arquiteturas *lakehouse* para dados heterogêneos [9, 10].

Se a hierarquia for válida, o dado é persistido na Bronze, o evento de ingestão é registrado e o *lineage* é atualizado. Em seguida, o pipeline aplica limpeza e padronização, executa checagens de qualidade e, em caso de aprovação, aciona etapas de reconstrução e derivação. As séries são então salvas na Silver e os agregados e indicadores são promovidos à Gold, com atualização de visões e publicação via APIs e *dashboards*. Registros reprovados permanecem rastreáveis e seguem para reprocessamento controlado.

4.8.5 Diagrama de sequência (interações entre módulos na ingestão)

A Figura 4.17 mostra o diagrama de sequência da ingestão de dados no FIDLER, evidenciando as interações entre os principais módulos: *Operador*, *Portal FIDLER*, *API de Ingestão*, *Orquestrador de Pipelines*, *Armazenamento Bronze* e *Catálogo de Metadados*.

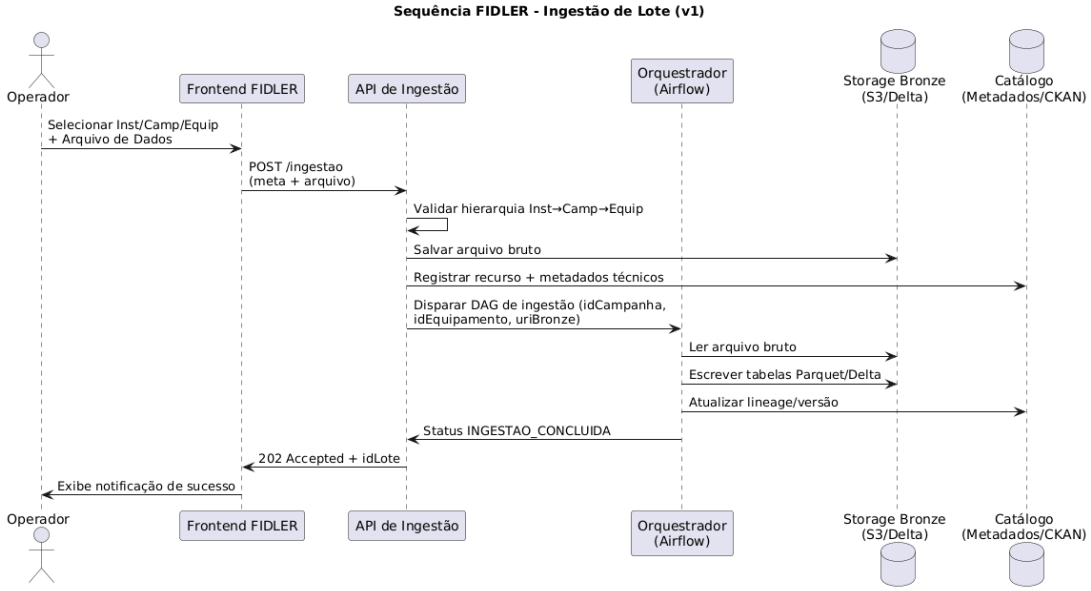


Figura 4.17 – Diagrama de sequência do FIDLER para ingestão de dados: interação entre operador, portal, API de ingestão, orquestrador de pipelines, armazenamento Bronze e catálogo de metadados. Elaborado pelo autor com base em arquiteturas transacionais de *Data Lakehouse* [5, 10].

O operador inicia a sequência via portal, que envia uma requisição `POST /ingestao` para a API. A API valida a hierarquia de domínio, escreve o arquivo bruto na Bronze, registra o recurso no catálogo interno e dispara um *job* no orquestrador. O orquestrador lê o arquivo da Bronze, gera tabelas Parquet/Delta, aplica transformações iniciais e atualiza o *lineage*. Ao final, o status da ingestão é retornado à API e, em seguida, ao portal, que exibe mensagem de sucesso ou falha para o operador. Este diagrama deixa explícito o encadeamento assíncrono entre interface, serviços e camada de dados.

4.8.6 Diagrama de componentes (arquitetura técnica do FIDLER)

A Figura 4.18 apresenta o diagrama de componentes do FIDLER, organizado em camadas de apresentação, serviços, processamento e dados, além de ferramentas de apoio à engenharia. Na apresentação, o Frontend React centraliza a interação com usuários. Em serviços, a API Go (REST) e o Swagger/OpenAPI concentram a exposição de endpoints e a documentação. Em processamento, os *pipelines* Python executam ETL e IA. Em dados, MinIO e PostgreSQL suportam armazenamento e consultas.

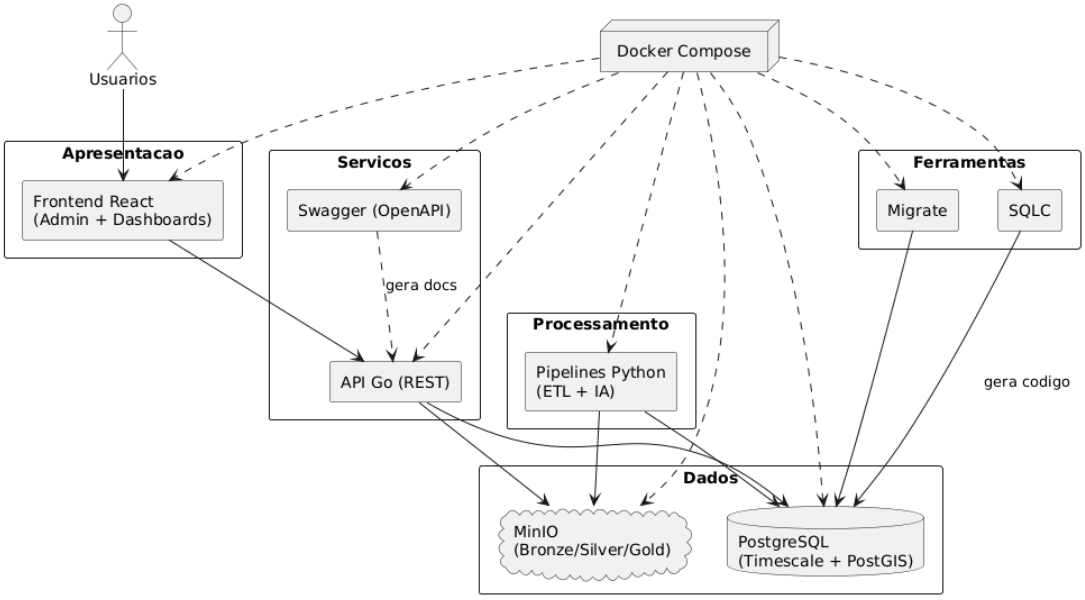


Figura 4.18 – Diagrama de componentes do FIDLER, com camadas de apresentação, serviços, processamento (pipelines) e dados (MinIO com Bronze/Silver/Gold e PostgreSQL). Elaborado pelo autor com base em arquiteturas modulares de *Data Lakehouse* [7, 8, 11].

A camada de processamento é representada pelos *pipelines* Python (ETL e IA), responsáveis por executar transformações Bronze–Silver–Gold e etapas de modelagem. Na camada de dados, destacam-se o MinIO como armazenamento das zonas Bronze/Silver/Gold e o PostgreSQL (Timescale/PostGIS) para consultas rápidas e geoespaciais. O diagrama também evidencia a camada de serviços (API Go REST e documentação via Swagger/OpenAPI), a apresentação (Frontend React) e as ferramentas de apoio (Migrate e SQLC), todas orquestradas via Docker Compose, refletindo as decisões das Tabelas 4.4 e 4.5.

4.8.7 Diagrama de estados (ciclo de vida dos dados)

A Figura 4.19 apresenta o diagrama de estados do ciclo de vida de um dado temporal no FIDLER. O estado inicial *Coletado* representa a medição realizada pelo

equipamento em campo. Quando o arquivo correspondente é submetido ao FIDLER, o registro passa ao estado *Ingerido*, associado à persistência na Bronze. Em seguida, o dado entra no estado *Validando*, no qual são aplicadas regras de integridade, faixas físicas, consistência temporal e coerência espacial.

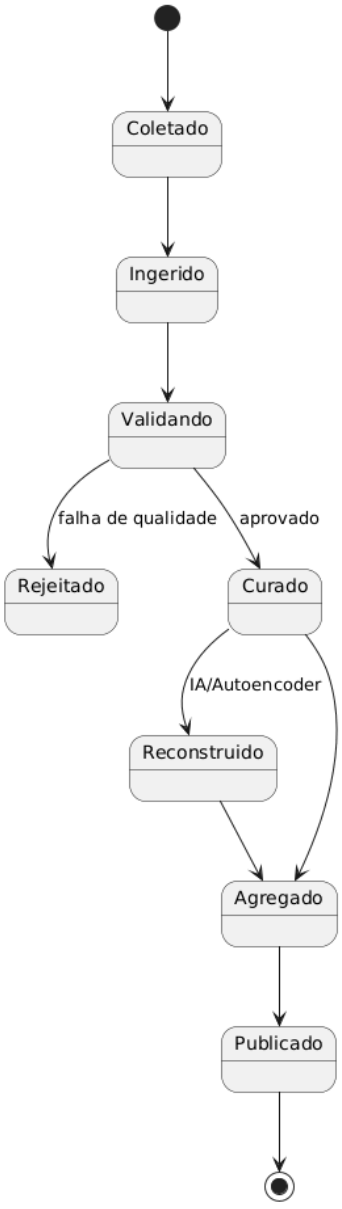


Figura 4.19 – Diagrama de estados do FIDLER para o ciclo de vida de um dado temporal: Coletado, Ingerido, Validando, Curado, Reconstruído, Agregado e Publicado. Elaborado pelo autor a partir de práticas de qualidade e reconstrução de séries temporais [9, 12].

Registros que falham nessas verificações são enviados ao estado *Rejeitado*, mantendo, contudo, rastreabilidade via *lineage*. Registros aprovados transitam para *Curado* (Silver) e, quando submetidos a interpolação, preenchimento de lacunas ou reconstrução via IA, podem entrar em *Reconstruído*. Após agregações temporais e derivação de indicadores,

os dados alcançam o estado *Agregado*, sendo finalmente promovidos a *Publicado*, quando ficam disponíveis para consultas, dashboards e relatórios. Este ciclo de vida está alinhado às práticas de qualidade e automação incremental de séries temporais energéticas [9, 12].

4.8.8 Diagrama de arquitetura do FIDLER (síntese)

Por fim, a Figura 4.20 apresenta o diagrama de arquitetura do FIDLER, sintetizando em uma única visão UML os elementos discutidos ao longo do capítulo: domínio institucional, arquitetura em camadas, metamodelos de dados, componentes técnicos e fluxos operacionais. O diagrama integra três blocos: (i) *Domínio FIDLER*, com instituições, campanhas e equipamentos; (ii) *Núcleo Lakehouse FIDLER*, com zonas Bronze, Silver e Gold e módulos de ingestão, qualidade, governança e IA/ML; e (iii) *Camada de Consumo*, com portal, APIs externas, *dashboards* e relatórios.

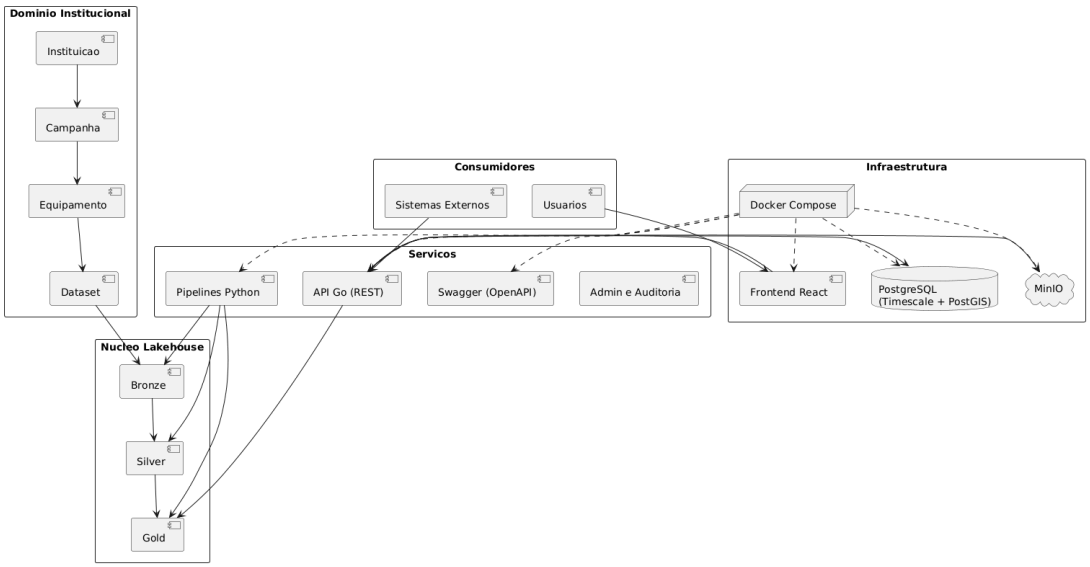


Figura 4.20 – Diagrama de arquitetura do FIDLER, integrando domínio institucional (instituições, campanhas, equipamentos), núcleo *lakehouse* (Bronze, Silver, Gold, governança, IA/ML) e camada de consumo (portal, APIs, *dashboards*, relatórios). Elaborado pelo autor a partir das estruturas conceituais discutidas no Capítulo 3.

As setas de dependência e composição ligam explicitamente instituições a campanhas, campanhas a equipamentos, equipamentos a datasets e datasets às zonas Bronze–Silver–Gold, respeitando a regra de que não existem dados soltos no sistema. Ao mesmo tempo, o diagrama mostra como requisitos de interoperabilidade, governança, latência, IA/ML e resiliência (Tabela 4.2) se materializam em componentes e fluxos concretos dentro do FIDLER.

4.8.9 Arquitetura *Lakehouse* de Referência (visão ampliada)

A Figura 4.21 apresenta uma visão ampliada da arquitetura *Lakehouse* que orientou o desenho do FIDLER. O diagrama explicita o fluxo ponta a ponta: fontes de dados heterogêneas (sensores e bases institucionais) alimentam a camada de ingestão, que registra metadados e versionamento antes do armazenamento bruto na Bronze. A partir daí, os *pipelines* promovem limpeza, padronização temporal e validações de qualidade para a Silver, onde os dados curados ficam prontos para análises consistentes. Na Gold, ocorre a derivação de indicadores, agregações e produtos analíticos, que são expostos por mecanismos de acesso (SQL, APIs e ferramentas analíticas). A figura também evidencia os pilares transversais de governança, segurança, *lineage* e orquestração, essenciais para rastreabilidade, reprodutibilidade e conformidade. Essa visão ampliada sintetiza a arquitetura seguida neste trabalho e esclarece como o FIDLER materializa cada bloco com tecnologias e decisões descritas ao longo do capítulo.

4.9 Síntese

Este capítulo aprofundou o desenvolvimento do FIDLER, conectando requisitos funcionais (Tabela 4.1) e não funcionais (Tabela 4.2), arquitetura em camadas, metamodelos de dados, componentes tecnológicos e fluxos operacionais. A partir das lacunas identificadas no Capítulo 3, foram priorizados requisitos de interoperabilidade, governança, latência analítica, suporte a IA/ML, resiliência e segurança (Tabela 4.2), materializados em uma arquitetura *lakehouse* Bronze–Silver–Gold (Figuras 4.1–4.7) e em decisões concretas de tecnologia (Tabela 4.4).

Os metamodelos de dados (Tabela 4.3) e o diagrama de classes (Figura 4.15) garantem que nenhuma instância de dado exista fora da hierarquia Instituição → Campanha → Equipamento → Dataset, assegurando rastreabilidade e aderência a princípios FAIR. O conjunto de diagramas UML – conceitual, casos de uso, atividades, sequência, componentes, estados e diagrama genérico final – oferece uma visão coerente e complementar do FIDLER, desde o contexto de alto nível até o comportamento dinâmico dos pipelines e o ciclo de vida dos dados.

Foram ainda detalhados mecanismos de qualidade, governança e segurança, bem como a integração de modelos de previsão e manutenção preditiva, alinhados aos trabalhos recentes em *Data Lakehouse* e séries temporais energéticas [9, 10, 12, 48–51]. Como próximos passos, propõe-se validar os SLOs definidos na Tabela 4.9 em ambientes de teste, refinar a implementação dos componentes técnicos descritos no diagrama de componentes (Figura 4.18) e utilizar o diagrama de arquitetura (Figura 4.20) como referência para a implementação incremental do FIDLER em ambientes reais de campanhas eólicas e

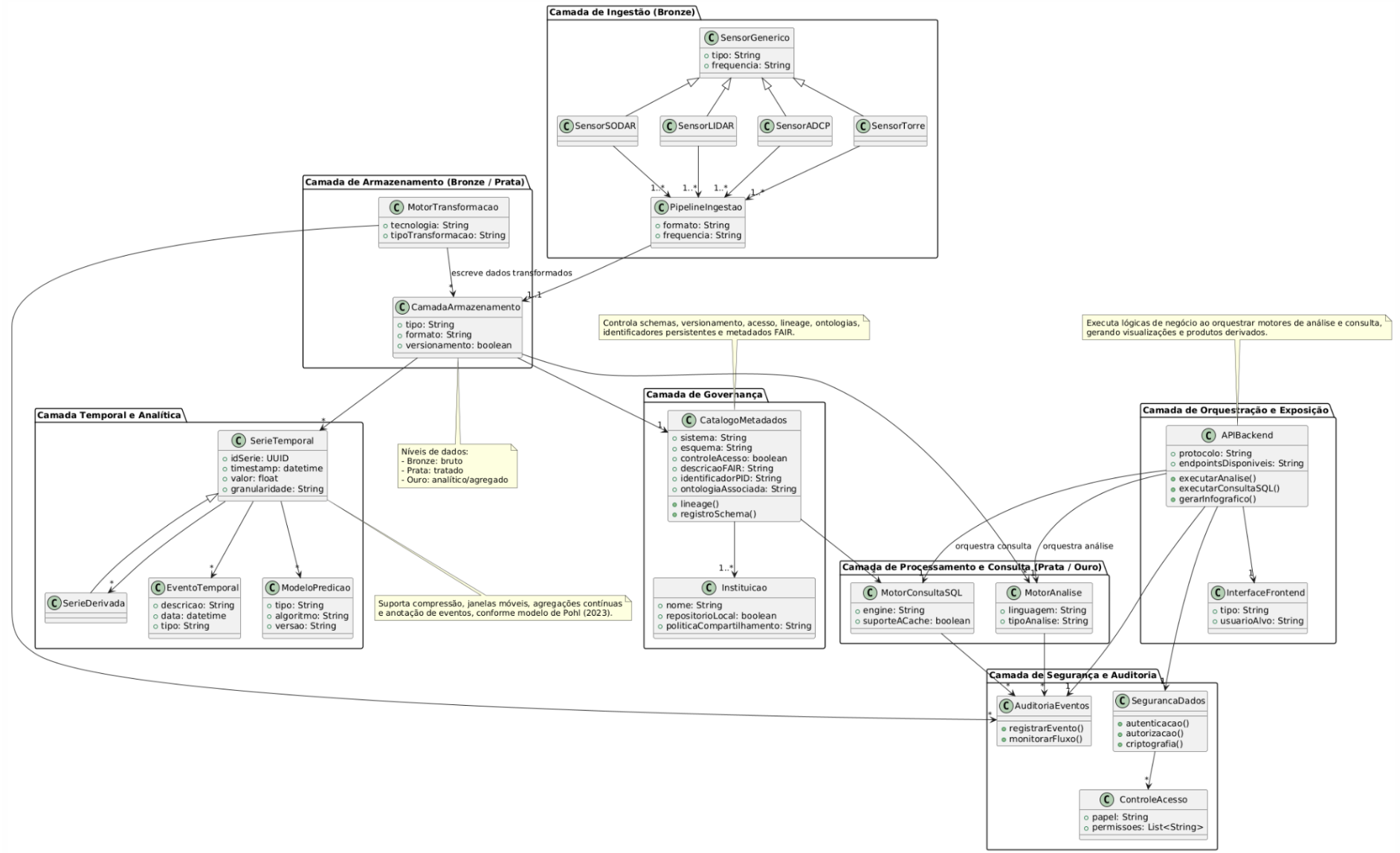


Figura 4.21 – Arquitetura *Data Lakehouse* de referência em visão ampliada, destacando o fluxo de ingestão, as camadas Bronze–Silver–Gold, os mecanismos de governança e as interfaces de consumo analítico.

maremotrizes.

5 EXEMPLOS ILUSTRATIVOS DE APLICAÇÕES DESENVOLVIDAS COM O *FRAMEWORK* FIDLER

Este capítulo apresenta exemplos ilustrativos da implementação do *framework*, com base em telas, diagramas e registros do protótipo desenvolvido. As figuras inseridas a seguir descrevem o ciclo de vida dos dados, a organização por zonas no *lakehouse*, o processo de ingestão das fontes e as principais interfaces do sistema (autenticação, operação, administração e auditoria), além da documentação de API e da infraestrutura de implantação.

5.1 Ciclo de Vida dos Dados e Visão Arquitetural

A Figura 5.1 apresenta a visão macro do ciclo de vida dos dados no *framework*, enfatizando os principais pontos de passagem entre coleta, ingestão, transformação e consumo.

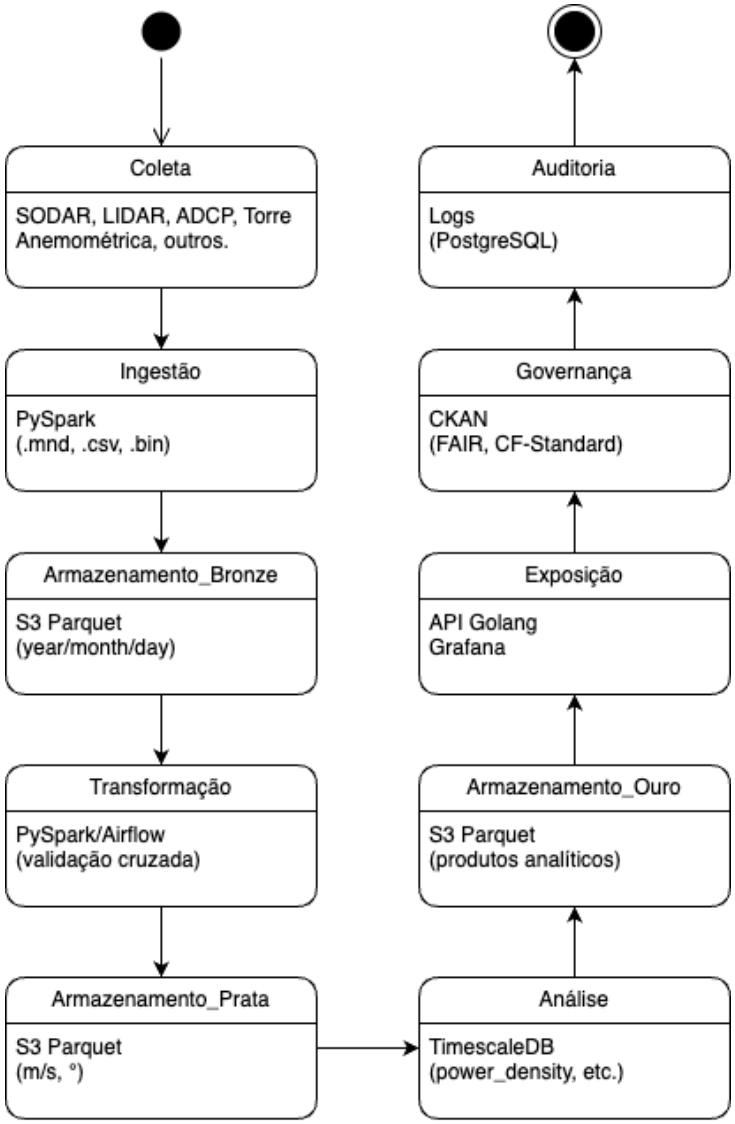


Figura 5.1 – Visão macro do ciclo de vida dos dados no *framework*.

O diagrama explicita a sequência de etapas à esquerda, iniciando com a coleta de dados por diferentes sensores e avançando para ingestão, armazenamento Bronze, transformação e armazenamento Prata, além de uma etapa de análise que alimenta produtos finais. À direita, aparecem os blocos de exposição, governança e auditoria, indicando que a disponibilização dos resultados se dá por APIs e *dashboards*, com controle de metadados e trilhas de auditoria.

A organização em duas colunas separa o fluxo operacional do fluxo de controle, indicando que qualidade, governança e auditoria não são atividades pontuais, mas camadas permanentes ao longo do ciclo. Essa representação justifica a necessidade de metadados, *lineage* e validações cruzadas como requisitos estruturantes do *framework*.

5.2 Lakehouse e Organização por Zonas

A Figura 5.2 introduz a visão geral do *lakehouse* adotado no protótipo, com a separação funcional entre camadas de dados e os principais componentes de processamento e consumo.

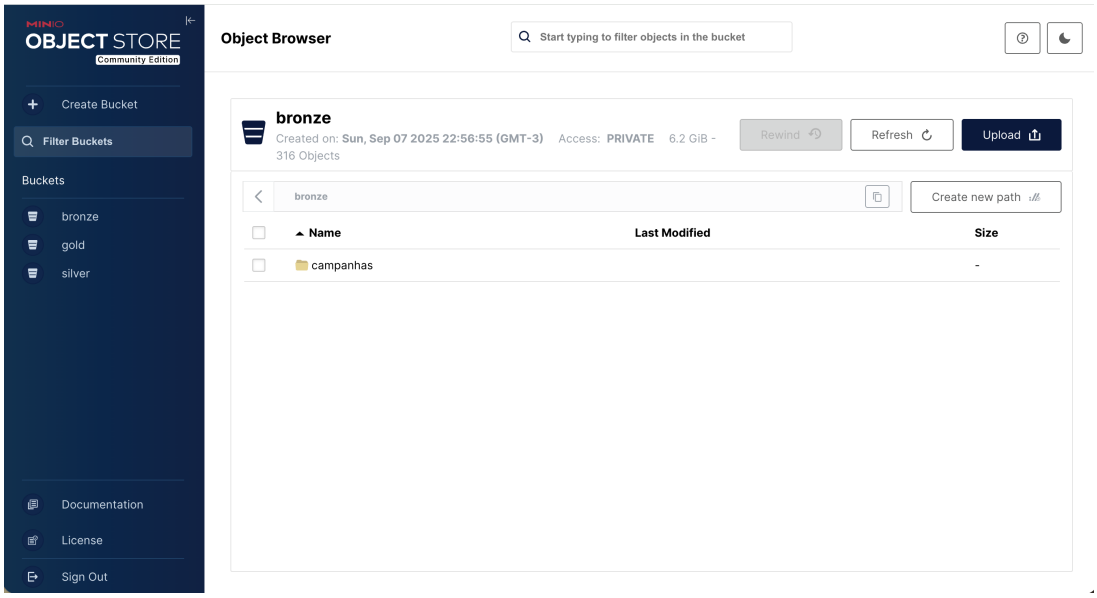


Figura 5.2 – Arquitetura *lakehouse* utilizada no protótipo.

O diagrama apresenta o fluxo central que conecta ingestão, transformação e consumo, indicando que a passagem entre camadas ocorre de forma ordenada. Observa-se a presença de mecanismos de governança e auditoria associados ao ciclo operacional, reforçando o controle sobre qualidade e uso dos dados.

Essa visão geral é relevante para contextualizar os exemplos posteriores, pois explicita a lógica de evolução dos dados no *lakehouse*. A leitura conjunta das camadas deixa claro que dados brutos e produtos analíticos coexistem, mas com níveis de curadoria distintos, requisito essencial para análises confiáveis.

A Figura 5.3 apresenta a zona Bronze como estágio de recepção de dados brutos, destacando a preservação do conteúdo original.

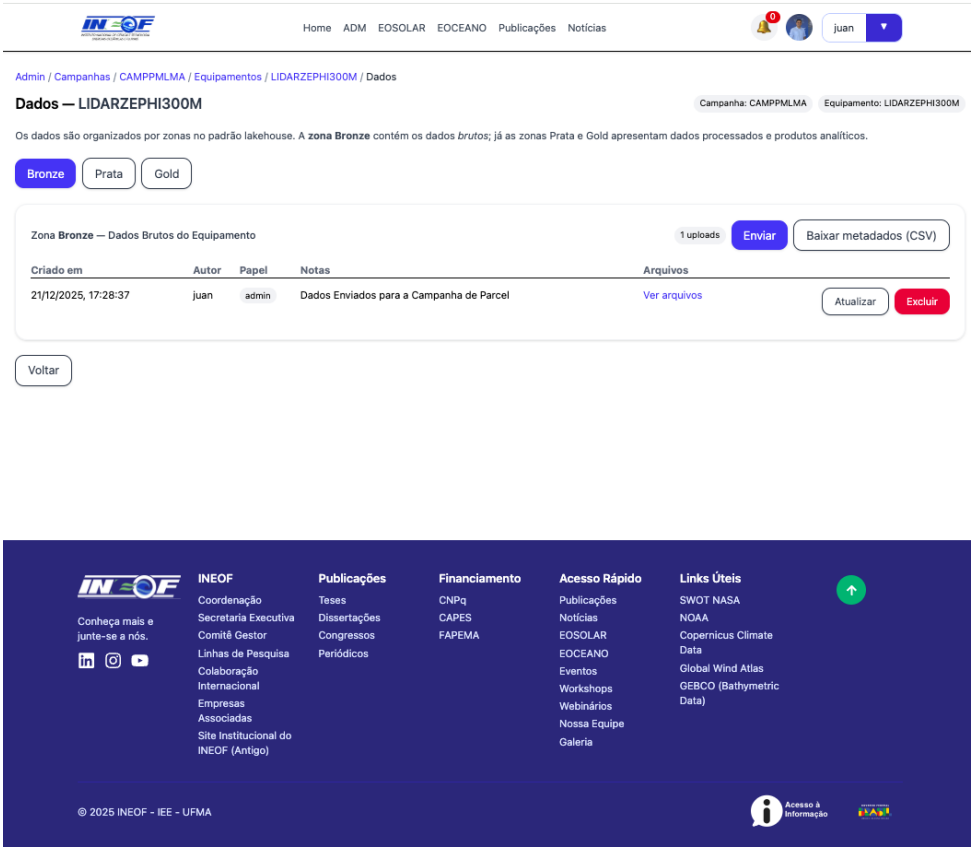


Figura 5.3 – Zona Bronze: dados brutos com rastreabilidade de origem.

Observa-se que a zona Bronze está orientada à *landing zone*, com retenção de arquivos originais e identificação da origem dos dados. Essa estrutura favorece reprocessamentos futuros, pois mantém o dado íntegro e rastreável.

O papel dessa camada é assegurar integridade e completude antes de qualquer curadoria. Por isso, o registro de metadados de ingestão e a organização por campanhas tornam-se fundamentais para garantir reprodutibilidade e auditoria.

A Figura 5.4 introduz a zona Silver, dedicada à padronização e enriquecimento das séries temporais.

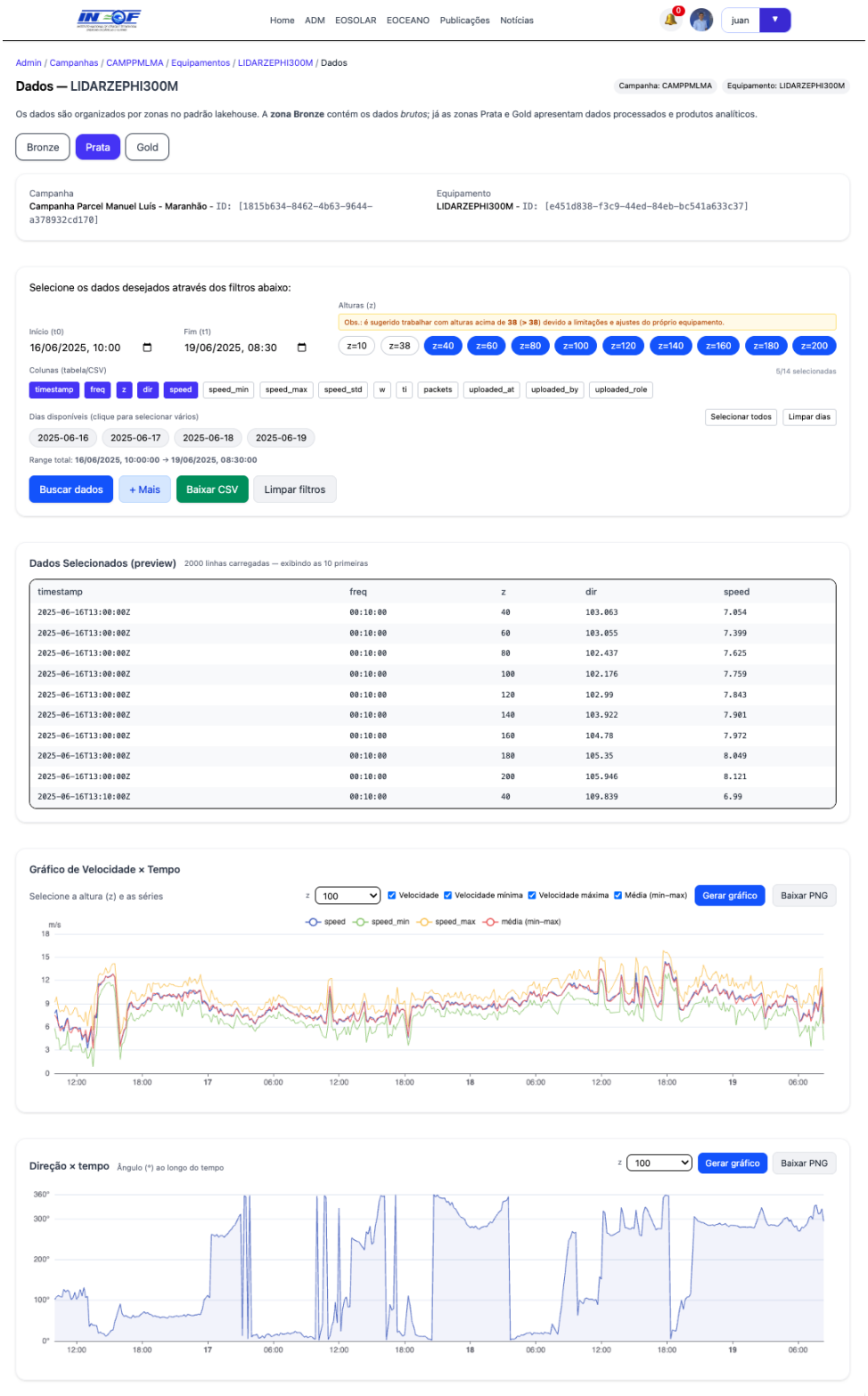


Figura 5.4 – Zona Silver: dados padronizados e enriquecidos.

No diagrama, a Silver aparece como etapa intermediária de curadoria, concentrando limpeza, validações e normalização. A ênfase está na consistência temporal e na padronização de unidades, condições indispensáveis para comparação entre campanhas e

equipamentos.

Essa camada prepara os dados para análises posteriores e reduz a propagação de erros. A existência de registros de qualidade e de parâmetros de transformação permite rastrear alterações e justificar decisões analíticas.

A Figura 5.5 apresenta a zona Gold, orientada ao consumo analítico e à entrega de produtos consolidados.

Figura 5.5 – Zona Gold: dados consolidados para análise e tomada de decisão.

O destaque da Gold está na geração de indicadores e séries derivadas, com foco em métricas energéticas e confiabilidade. A organização sugere que os dados já foram

avaliados e preparados para consumo por diferentes perfis de usuário.

Esse nível é o ponto de contato com *dashboards*, APIs e relatórios. Assim, a Gold opera como camada de prestação de serviço, garantindo que decisões sejam tomadas com base em dados curados e contextualizados.

A Figura 5.6 introduz um exemplo de transformação entre Bronze e Silver, contrastando dados brutos e dados padronizados.

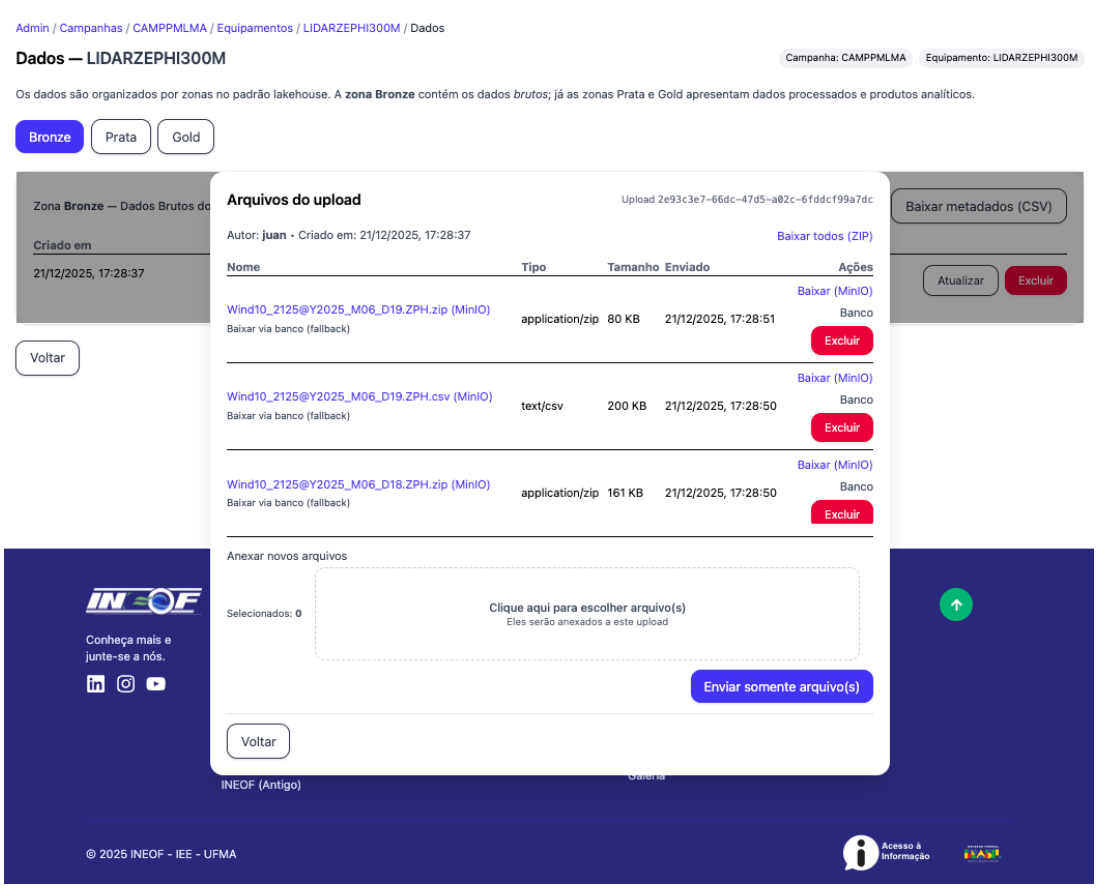


Figura 5.6 – Exemplo de dados antes e depois da padronização (Bronze → Silver).

O quadro evidencia diferenças de estrutura e padronização, tornando visíveis inconsistências removidas e campos normalizados. A apresentação lado a lado facilita a identificação do efeito da curadoria.

Esse exemplo é relevante para justificar o investimento na camada Silver. A transformação reduz ruído, melhora a legibilidade das séries e cria condições para análises estatísticas e energéticas mais robustas.

A Figura 5.7 detalha a comparação entre dados brutos e curados em um caso real.

Na comparação, observa-se a reorganização de colunas, a padronização de unidades e a eliminação de campos inconsistentes. A estrutura Silver apresenta maior regularidade temporal e semântica, favorecendo consultas e análises sistemáticas.

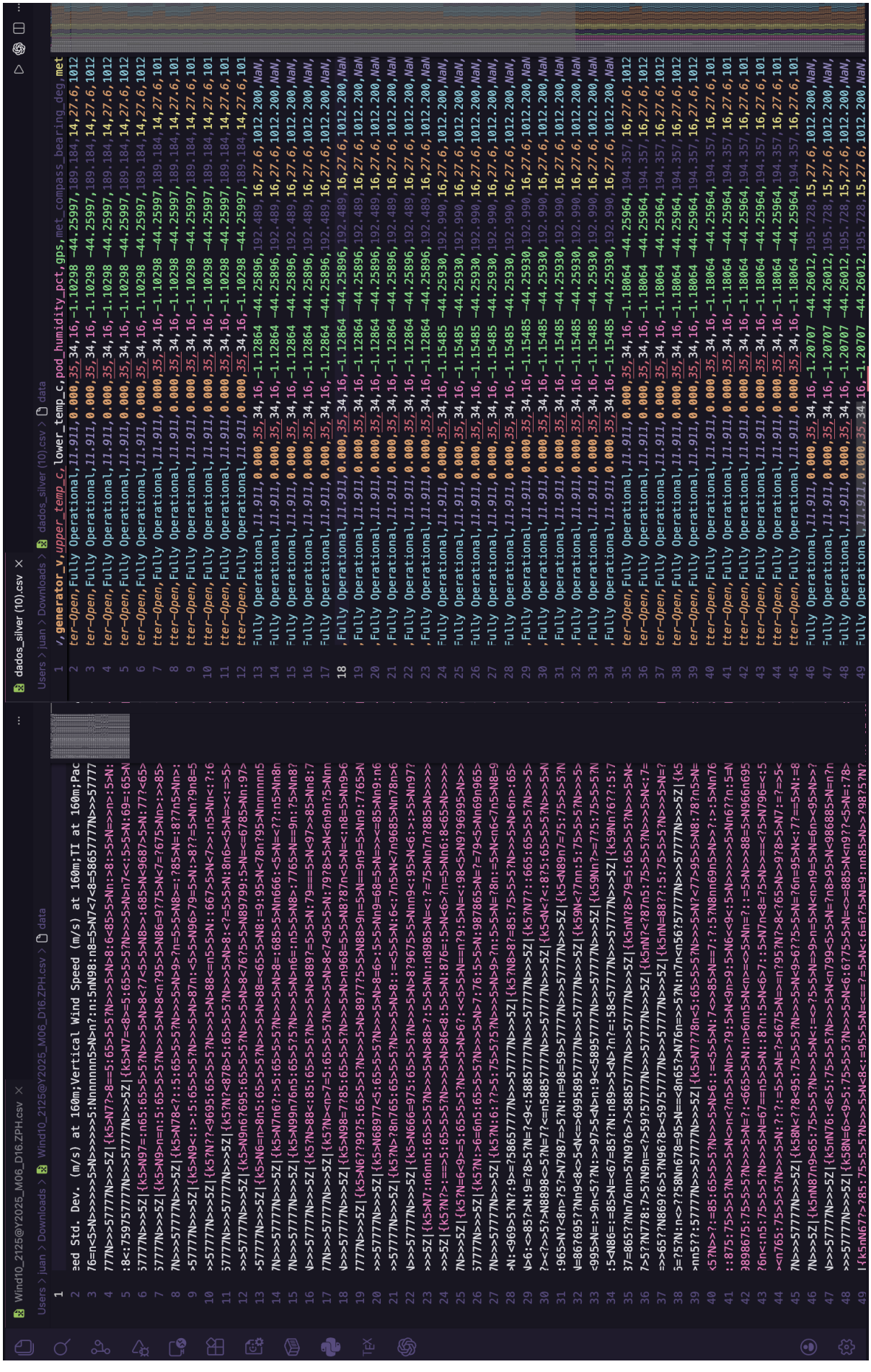


Figura 5.7 – Comparação detalhada entre dados Bronze e Silver para um caso de estudo.

O caso ilustra a redução de ambiguidade e a melhoria da qualidade informacional. Ao padronizar os dados, o *framework* garante maior comparabilidade entre campanhas e equipamentos distintos.

A Figura 5.8 apresenta um produto derivado na zona Silver, com o cálculo de REWS.

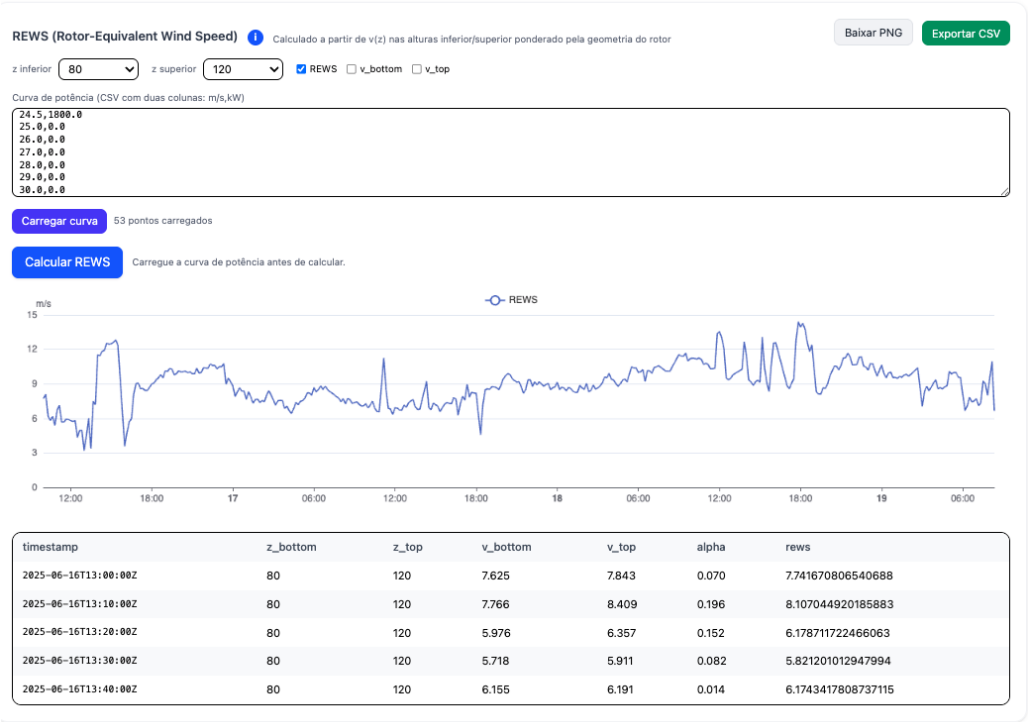


Figura 5.8 – Resultado de cálculo de REWS na zona Silver.

O gráfico e a tabela associados evidenciam a aplicação do cálculo de *Rotor-Equivalent Wind Speed*, com parâmetros de altura inferior e superior e a curva de potência carregada. A interface também oferece exportação de resultados, indicando que o derivado já pode ser reutilizado em análises posteriores.

Esse exemplo mostra como a Silver não apenas limpa dados, mas também produz variáveis físicas relevantes para a engenharia. O cálculo de REWS funciona como etapa de enriquecimento que prepara a série para uso em previsões e indicadores na Gold.

5.3 Ingestão e Integração das Fontes

A Figura 5.9 apresenta a integração entre fontes oceanográficas e anemométricas no *pipeline*, destacando a coexistência de variáveis com naturezas distintas.

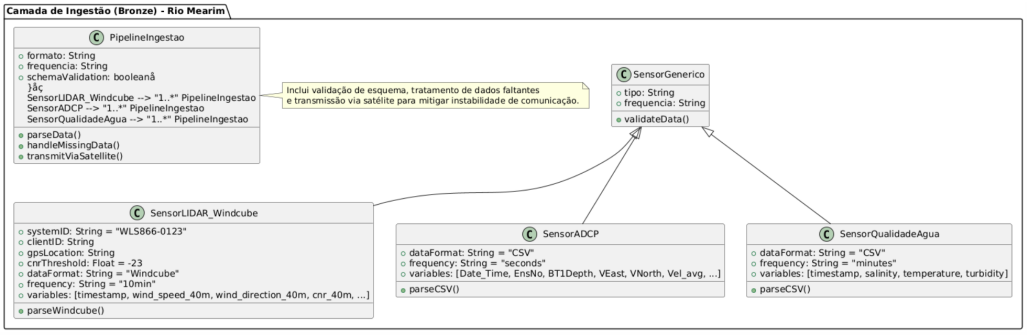


Figura 5.9 – Integração entre dados de ADCP e LIDAR no *pipeline*.

O diagrama evidencia a combinação de medições de corrente (ADCP) e vento (LIDAR), que passam a compor um histórico integrado. Essa integração viabiliza análises que relacionam dinâmica oceânica e condições atmosféricas no mesmo intervalo temporal.

Esse acoplamento é relevante para estudos maremotrizes e eólicos, pois permite correlacionar variáveis físicas e reduzir lacunas de informação. A representação também sugere a necessidade de harmonização temporal e padronização de unidades antes da curadoria.

A Figura 5.10 introduz a integração entre sensores LIDAR e SODAR, com foco na complementaridade dos perfis de vento.

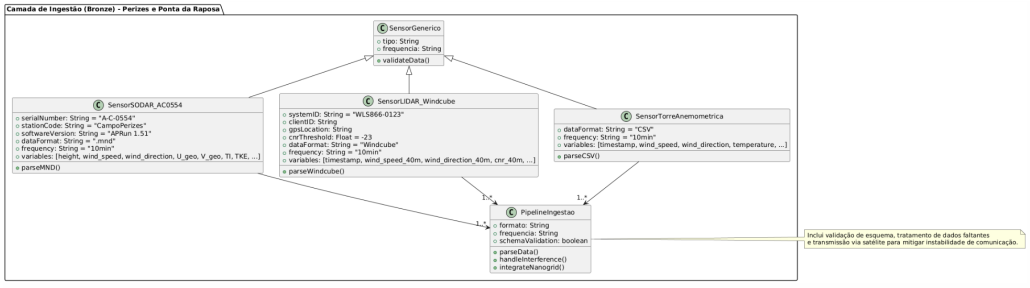


Figura 5.10 – Integração entre dados de LIDAR e SODAR.

O esquema evidencia que as duas tecnologias podem ser combinadas para ampliar a cobertura de alturas e frequências de medição. Essa integração reforça a robustez dos dados ao reduzir zonas de sombra e ampliar a densidade de observações.

Ao utilizar fontes distintas, o *framework* ganha redundância e capacidade de validação cruzada, o que melhora a confiabilidade das séries curadas. Essa estratégia também reduz vieses associados a um único tipo de sensor.

No protótipo, a ingestão pode ser realizada pelo administrador diretamente no *frontend*, enviando os arquivos para a zona Bronze. Essa opção garante controle institucional

sobre a entrada dos dados e facilita o registro de autoria, campanha e equipamento associados.

Para inspeção visual e organização dos *buckets*, recomenda-se o uso do console do MinIO em `http://localhost:9001`. A interface gráfica permite verificar a hierarquia de diretórios, conferir tamanhos e datas de atualização e realizar validações básicas de integridade.

Evita-se o uso direto de operações S3 manuais por reduzir erros operacionais e divergências de versionamento. A recomendação fortalece a governança e mantém a ingestão alinhada ao fluxo definido no *framework*.

5.4 Interfaces do Sistema e Governança

A Figura 5.11 apresenta a tela de autenticação do sistema, responsável por controlar o acesso às funcionalidades do *framework*.

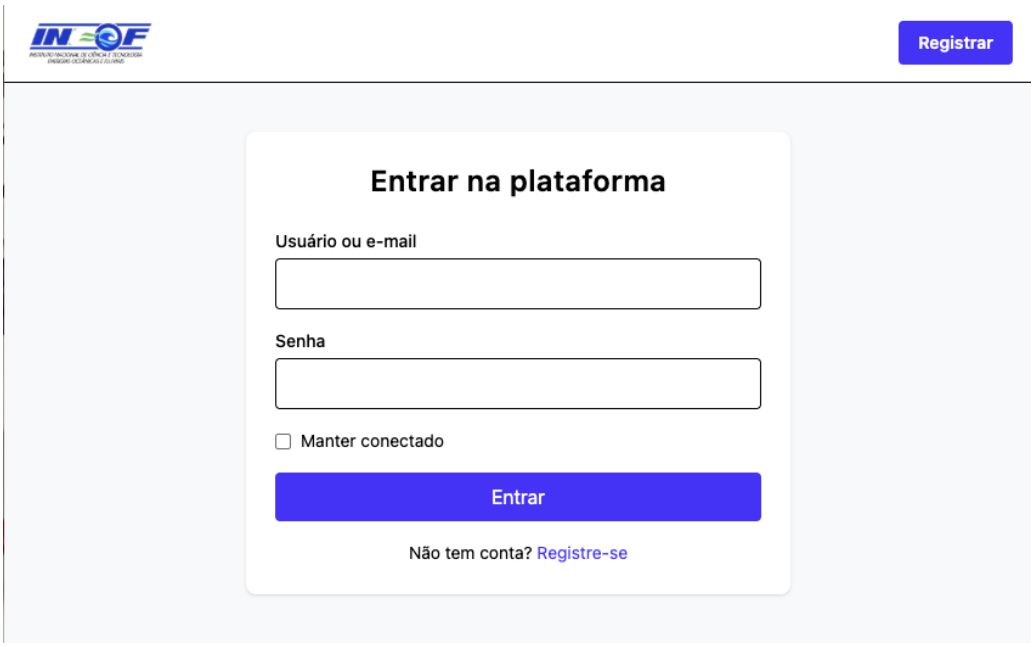


Figura 5.11 – Tela de login e autenticação de usuários.

A interface exibe campos de usuário/e-mail e senha, opção de manter a sessão ativa e vínculo para registro. Esses elementos indicam o controle de identidade e a separação entre usuários cadastrados e novos acessos.

O controle de acesso por perfil garante que apenas usuários autorizados realizem operações de ingestão, curadoria ou consulta. Dessa forma, a autenticação funciona como primeiro mecanismo de governança, evitando que dados sensíveis sejam manipulados sem permissão.

A Figura 5.12 introduz a tela inicial do ambiente, destacando a identidade institucional e a organização do conteúdo do portal.

Figura 5.12 – Tela inicial com visão geral do ambiente e indicadores.

Observa-se a presença do menu superior, navegação por seções e área de boas-vindas, além de blocos informativos sobre objetivos e linhas de pesquisa. A interface também exibe opções de conta do usuário, sugerindo integração entre conteúdo institucional e funcionalidades operacionais.

Essa página funciona como ponto de entrada operacional e institucional. Ela orienta o usuário no ecossistema do *framework* e facilita o acesso às áreas específicas de campanhas, equipamentos e dados.

A Figura 5.13 apresenta a área de gestão de campanhas e equipamentos, com informações descritivas e geográficas da campanha.

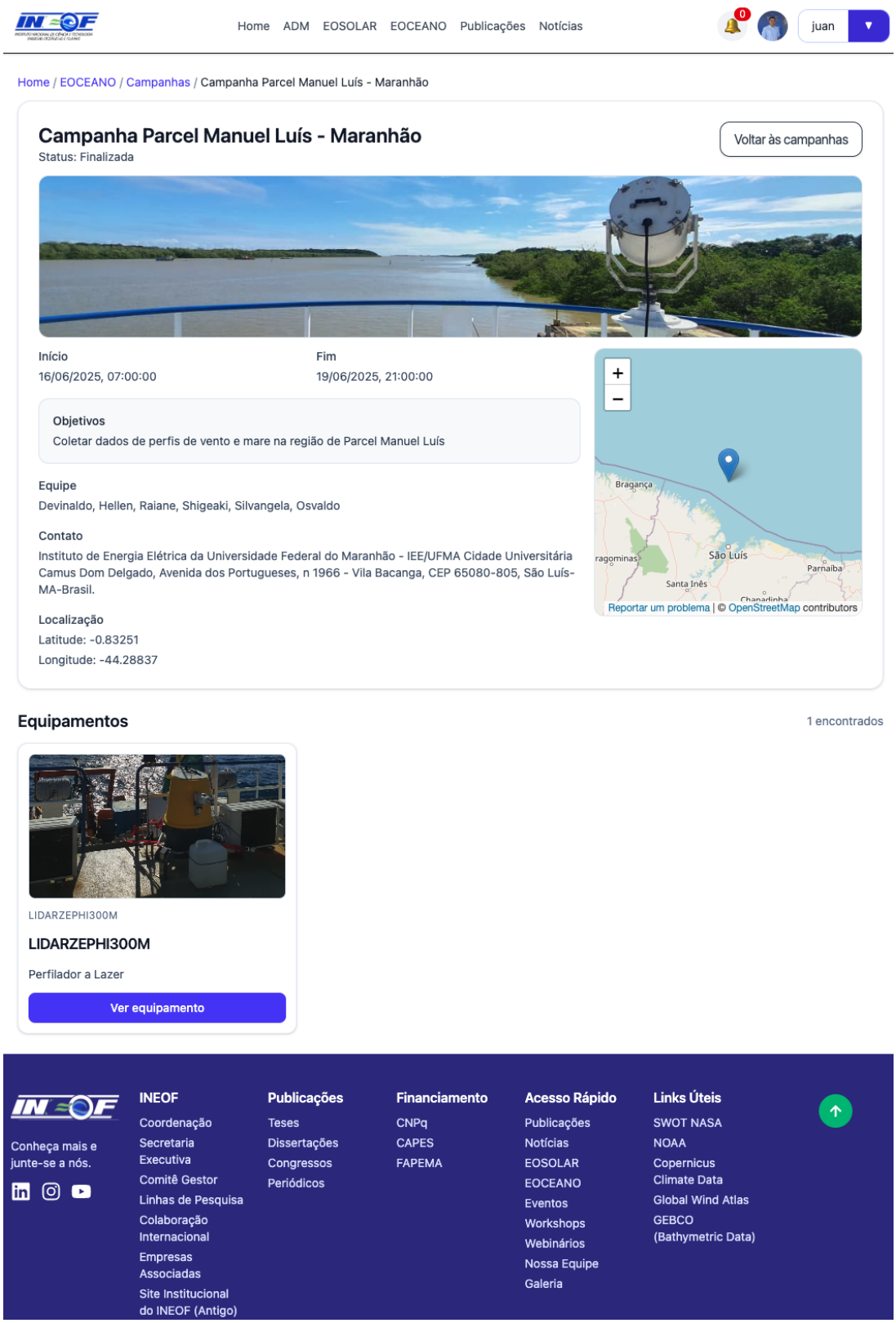


Figura 5.13 – Gestão de campanhas e equipamentos cadastrados.

A interface exibe dados de identificação da campanha, período, objetivos, equipe e localização geográfica, além de um mapa com o ponto de medição. Na seção inferior, são

listados os equipamentos associados, com acesso direto aos detalhes.

O cadastro organiza a hierarquia institucional, permitindo associar campanhas e equipamentos antes da ingestão de dados. Essa organização evita ambiguidades e garante que cada dataset seja corretamente vinculado ao contexto de coleta.

A Figura 5.14 apresenta o módulo administrativo de governança, reunindo funções de gestão e controle operacional.

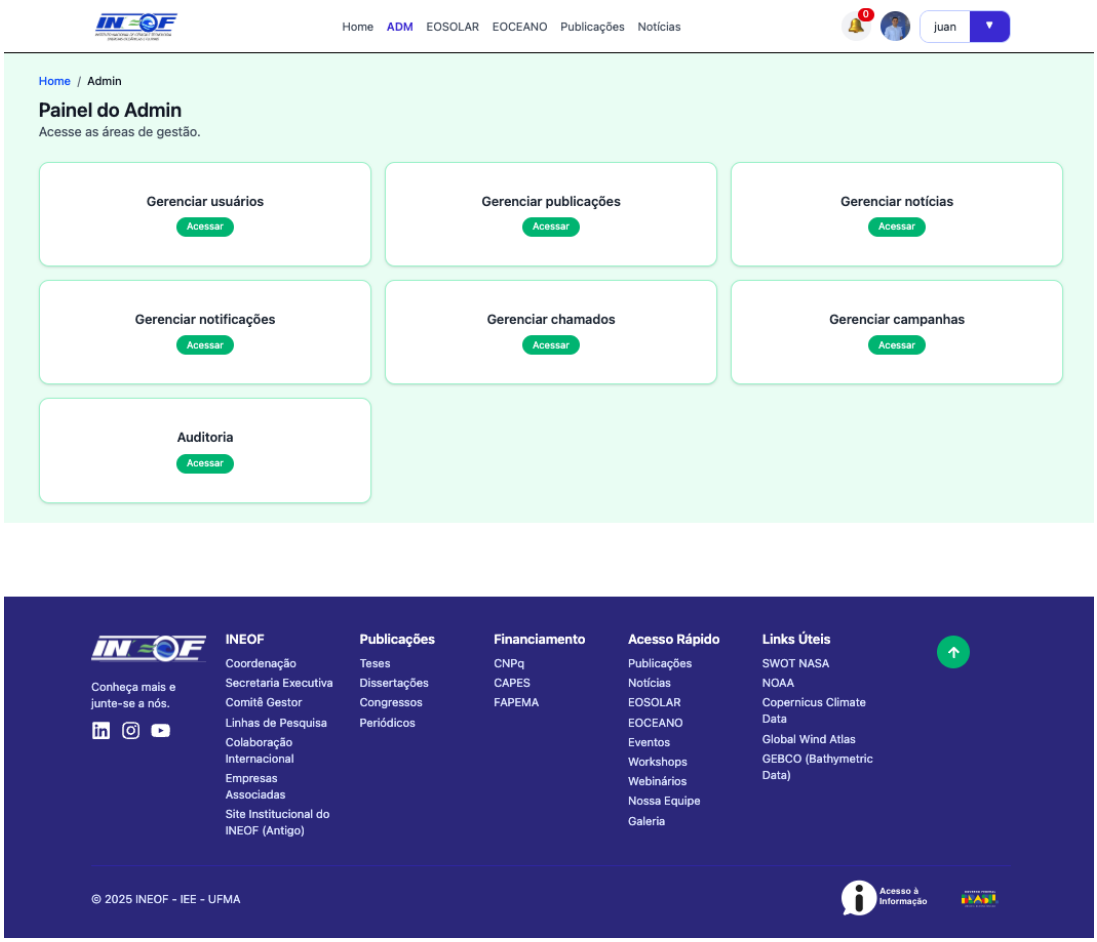


Figura 5.14 – Módulo administrativo para gestão de perfis e permissões.

Observa-se um painel com cartões de acesso para usuários, publicações, notícias, notificações, chamados, campanhas e auditoria. Essa organização indica a centralização das funções administrativas e a separação de responsabilidades.

O módulo centraliza perfis, permissões e recursos, assegurando conformidade e segregação de responsabilidades. A existência de atalhos específicos reduz o risco de operações indevidas e facilita a manutenção do ambiente.

A Figura 5.15 apresenta o registro de auditoria, com filtros e detalhamento de eventos.

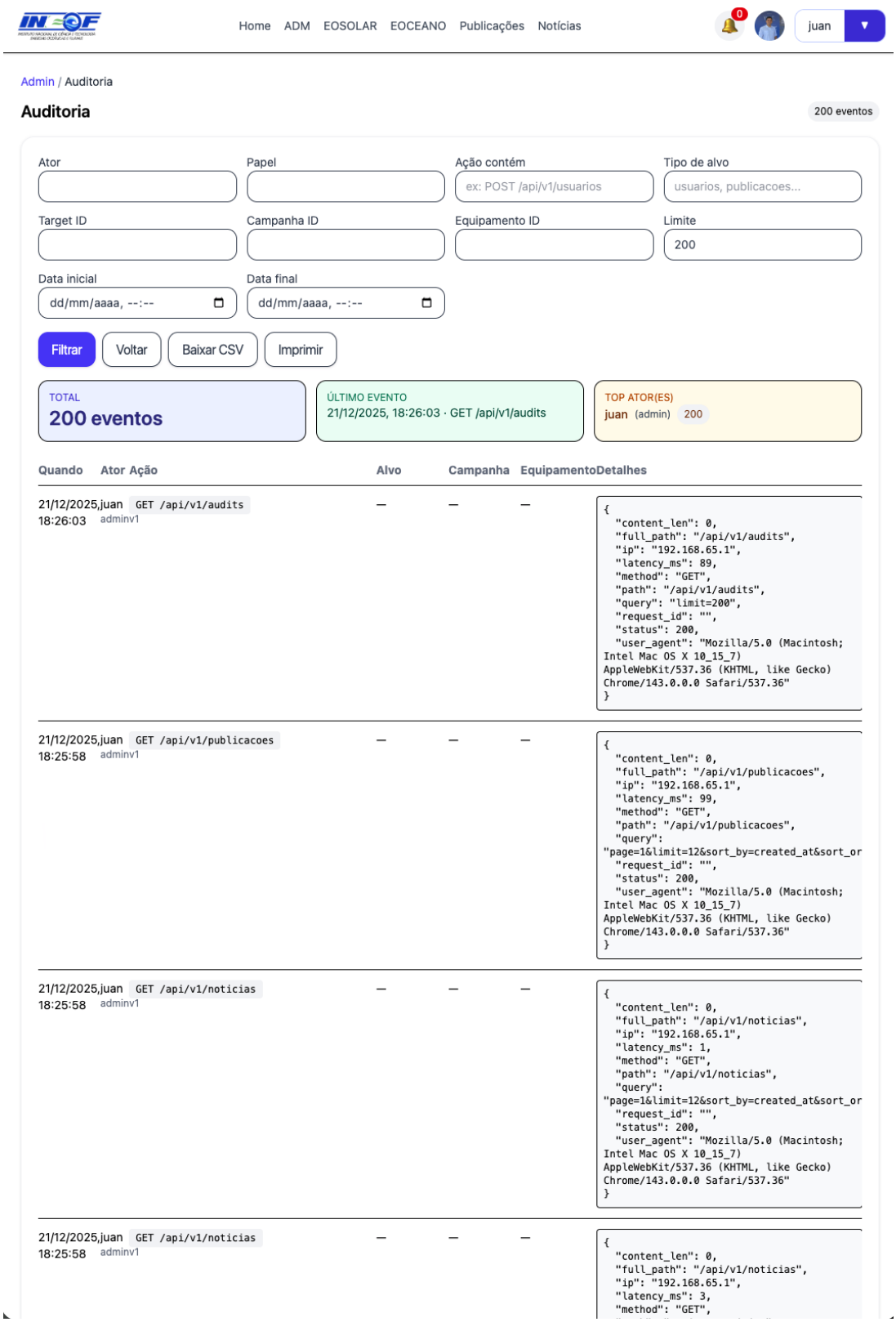


Figura 5.15 – Registro de auditoria com histórico de ações.

O painel exibe filtros por ator, tipo de ação, alvo e período, além de uma listagem cronológica de eventos com detalhes técnicos. Há também opções de exportação e impressão, indicando apoio à prestação de contas.

O histórico permite rastrear operações críticas, reforçando transparência e responsabilidade sobre alterações. Essa rastreabilidade é essencial para auditorias formais e para a verificação de conformidade em ambientes colaborativos.

A Figura 5.16 introduz a documentação interativa da API, utilizada para expor os contratos de serviço.

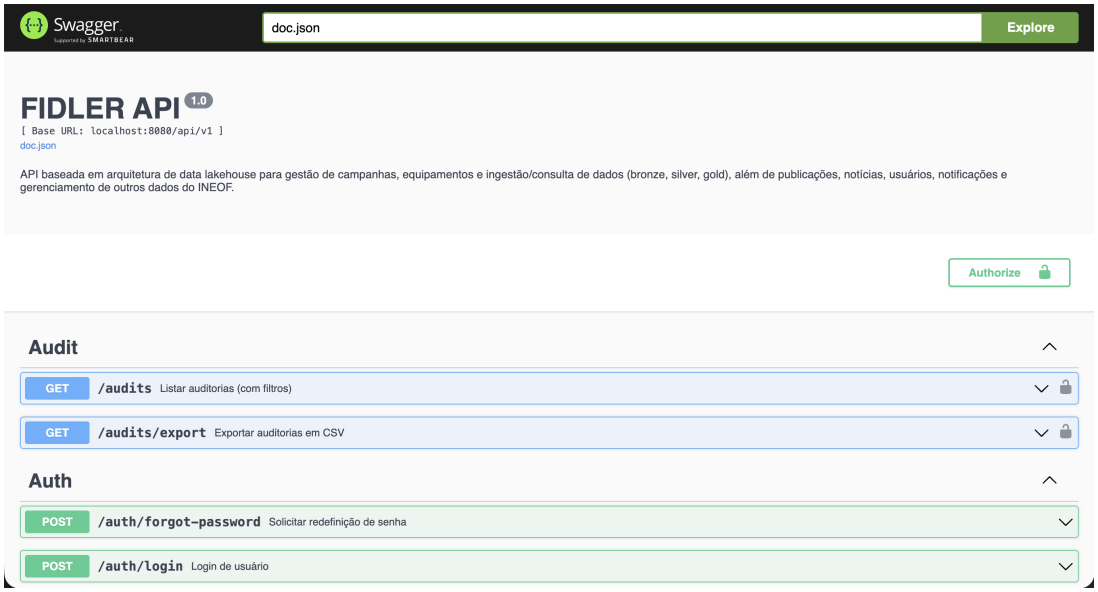


Figura 5.16 – Documentação interativa da API via Swagger.

O ambiente apresenta a lista de rotas organizadas por domínio funcional (por exemplo, auditoria e autenticação), com suporte a execução de testes e autenticação por token. A presença do arquivo `doc.json` indica geração automática de especificações.

Os *endpoints* são expostos com contratos e testes integrados, facilitando o consumo por sistemas externos. Essa documentação é fundamental para integração com ferramentas analíticas e para a padronização de acesso aos dados.

5.5 Infraestrutura de Implantação

A Figura 5.17 apresenta a visão geral da infraestrutura em contêineres do protótipo, organizada para facilitar implantação e repetibilidade do ambiente.

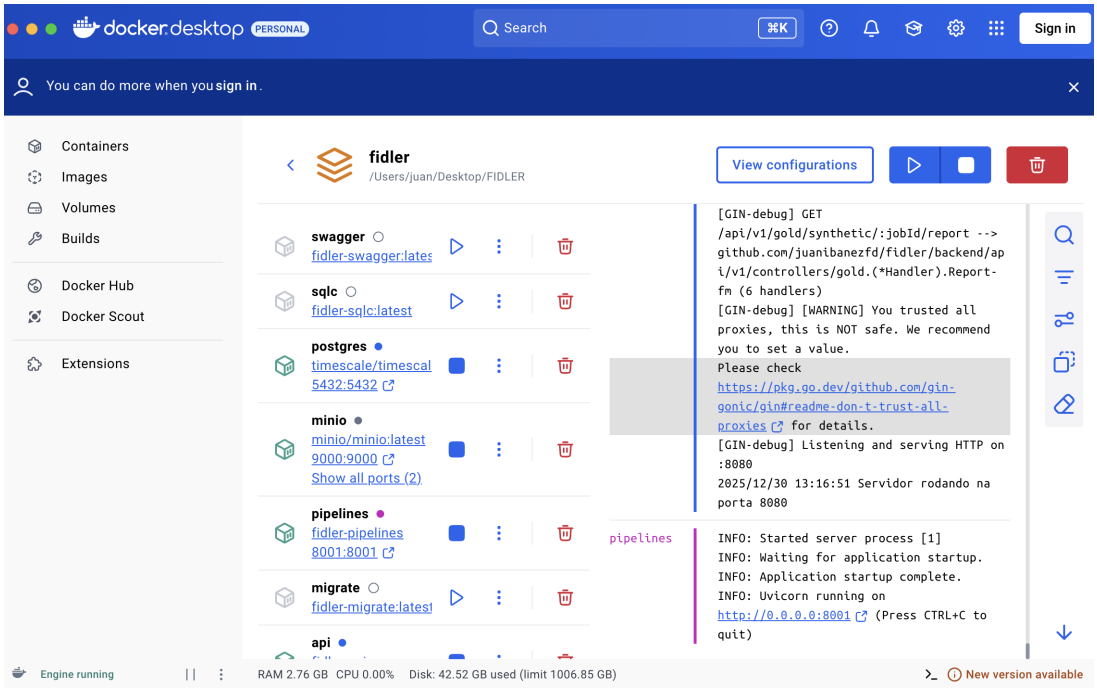


Figura 5.17 – Infraestrutura baseada em contêineres para execução dos serviços.

O painel do Docker Desktop lista os serviços ativos do ambiente, incluindo componentes principais (API, pipelines, banco e armazenamento) e serviços auxiliares (Swagger, SQLC e migrações). A visualização simultânea de *logs* confirma o funcionamento em tempo real e oferece visibilidade operacional sobre requisições e respostas.

Essa representação é relevante para demonstrar a execução integrada do protótipo em contêineres, garantindo isolamento e reprodutibilidade. Ao evidenciar as portas e serviços disponíveis, o ambiente facilita diagnósticos e validações durante o desenvolvimento.

A separação entre camadas de dados, processamento e serviços é reforçada pela distribuição em contêineres independentes. Essa estratégia reduz acoplamento, permite escalonamento seletivo e torna o ambiente mais controlável em testes e demonstrações.

O *docker-compose* coordena os serviços do protótipo, garantindo a inicialização ordenada e a comunicação entre banco de dados, armazenamento de objetos, API, *pipelines* e *frontend*. Para maior legibilidade, o arquivo `docker-compose.yml` divide-se em blocos funcionais, apresentados a seguir.

5.5.1 Banco de Dados (TimescaleDB + PostGIS)

O trecho a seguir apresenta o serviço de banco de dados temporal e geoespacial utilizado no protótipo.

```
postgres:
```

```
image: timescale/timescaledb-postgis:latest-pg12
container_name: fidler-db
restart: always
ports:

"5432:5432"
env_file:
.env.development
environment:
POSTGRES_USER: ${POSTGRES_USER:-postgres}
POSTGRES_PASSWORD: ${POSTGRES_PASSWORD:-postgres}
POSTGRES_DB: ${POSTGRES_DB:-fidler}
volumes:
pgdata:/var/lib/postgresql/data
networks:
fdlher-network
healthcheck:
test: ["CMD-SHELL", "pg_isready -U ${POSTGRES_USER:-postgres}"]
interval: 10s
timeout: 5s
retries: 5
```

Esse serviço armazena séries temporais e metadados espaciais, oferecendo suporte a consultas analíticas e operações georreferenciadas com persistência garantida por volume dedicado.

5.5.2 Armazenamento de Objetos (MinIO — Bronze/Silver/Gold)

O trecho a seguir define o serviço de armazenamento de objetos responsável pelas zonas Bronze, Silver e Gold.

```
minio:
image: minio/minio:latest
container_name: fidler-minio
command: server /data --console-address ":9001"
env_file:

.env.development
environment:
```

```
MINIO_ROOT_USER: ${MINIO_ROOT_USER:-fidleradmin}
MINIO_ROOT_PASSWORD: ${MINIO_ROOT_PASSWORD:-fidlersecret}
ports:
"9000:9000"  # API S3
"9001:9001"  # Console Web
volumes:
minio_data:/data
networks:
fdlher-network
healthcheck:
test: ["CMD", "curl", "-f", "http://localhost:9000/minio/health/live"]
interval: 30s
timeout: 20s
retries: 3
```

O MinIO fornece compatibilidade S3 e console web para inspeção de *buckets*, viabilizando a organização por zonas e o armazenamento de arquivos brutos e curados.

5.5.3 Backend — API em Go (MVC)

O bloco abaixo descreve o serviço da API responsável por ingestão, consulta e governança via endpoints REST.

```
api:
build:
context: ./backend/api/v1
dockerfile: Dockerfile
container_name: fidler-api
env_file:

.env.development
environment:
DB_DSN: postgres://${POSTGRES_USER:-
postgres}:${POSTGRES_PASSWORD:postgres}@postgres:5432/${POSTGRES_DB:-fidler}?
sslmode=disable
S3_ENDPOINT: http://minio:9000
S3_ACCESS_KEY: ${MINIO_ROOT_USER:-fidleradmin}
S3_SECRET_KEY: ${MINIO_ROOT_PASSWORD:-fidlersecret}
S3_BUCKET_BRONZE: bronze
```

```
S3_BUCKET_SILVER: silver
S3_BUCKET_GOLD: gold
depends_on:
postgres:
condition: service_healthy
minio:
condition: service_healthy
ports:
"8080:8080"
networks:
fdlher-network
```

O serviço centraliza regras de negócio e integra-se ao PostgreSQL e ao MinIO, garantindo acesso controlado às zonas e registro de metadados.

5.5.4 Pipelines Python — ETL e Inteligência Artificial

O trecho seguinte apresenta o serviço responsável pelos pipelines de transformação e IA.

```
pipelines:
  build:
    context: ./pipelines
    dockerfile: Dockerfile
  container_name: fidler-pipelines
  env_file:
    - .env.development
  environment:
    DB_DSN: postgresql://${POSTGRES_USER:-postgres}:${POSTGRES_PASSWORD:-postgres}@postgres:5432/${POSTGRES_DB:-fidler}
    PYTHON_DB_DSN: postgresql://${POSTGRES_USER:-postgres}:${POSTGRES_PASSWORD:-postgres}@postgres:5432/${POSTGRES_DB:-fidler}
    S3_ENDPOINT: http://minio:9000
    S3_ACCESS_KEY: ${MINIO_ROOT_USER:-fidleradmin}
    S3_SECRET_KEY: ${MINIO_ROOT_PASSWORD:-fidlersecret}
    S3_BUCKET_BRONZE: bronze
    S3_BUCKET_SILVER: silver
    S3_BUCKET_GOLD: gold
    SILVER_ROOT: /data/silver/campanhas
  volumes:
    - ./pipelines:/app
    - ./lakehouse:/data
  depends_on:
    postgres:
      condition: service_healthy
    minio:
      condition: service_healthy
  networks:
    - fdlher-network
```

Esse serviço executa curadoria, validações e processos de reconstrução, acessando o banco temporal e os *buckets* do *lakehouse*.

5.5.5 Frontend React — Administração e Dashboards

O bloco abaixo descreve o serviço do *frontend* utilizado para administração e visualização.

```
react:
build:
context: ./frontend/react-admin
dockerfile: Dockerfile
container_name: fidler-react
depends_on:

api
ports:
"5173:5173"
networks:
fdlher-network
```

A interface fornece acesso às funcionalidades operacionais e a painéis, comunicando-se com a API para autenticação e consulta.

5.5.6 Serviços Auxiliares (Migrações, Geração de Código e Documentação)

O trecho a seguir agrega serviços auxiliares para migração de banco, geração de código e documentação de API.

```
migrate:
image: fidler-migrate:latest
build:
context: ./backend/api/v1
dockerfile: Dockerfile.migrate
volumes:

./backend/api/v1/services/db/migration:/app/migration
command:
[
"-source",
"file:///app/migration",
"-database",
```

```
"postgres://postgres:postgres@fidler-db:5432/fidler?sslmode=disable",
"-verbose",
"up",
]
depends_on:
postgres:
condition: service_healthy
networks:
fdlher-network

sqlc:
build:
context: ./backend/api/v1
dockerfile: Dockerfile.sqlc
image: fidler-sqlc:latest
volumes:

./backend/api/v1:/app
profiles:
tools
networks:
fdlher-network

swagger:
build:
context: ./backend/api/v1
dockerfile: Dockerfile.swagger
image: fidler-swagger:latest
volumes:

./backend/api/v1:/app
profiles:
tools
networks:
fdlher-network
```

Esses serviços automatizam etapas de manutenção, garantindo consistência de esquema, geração tipada e documentação atualizada.

5.5.7 Volumes Persistentes e Rede Compartilhada

```
volumes:
pgdata:
minio_data:
networks:
fdlher-network:
driver: bridge
```

Em termos funcionais, o serviço `postgres` mantém dados transacionais e espaciais (TimescaleDB + PostGIS), o `minio` hospeda os objetos do *lakehouse* (Bronze/Silver/Gold), o `api` centraliza serviços REST, o `pipelines` executa ETL e inteligência artificial, e o `react` entrega o *frontend*. Os serviços auxiliares `migrate`, `sqlc` e `swagger` automatizam migrações de banco, geração de código type-safe e documentação OpenAPI, garantindo repetibilidade e rastreabilidade.

5.6 Síntese

Este capítulo apresentou o conjunto de telas e diagramas do protótipo, descrevendo como os dados percorrem o ciclo de vida, organizam-se em zonas no *lakehouse*, ingerem-se a partir de sensores e disponibilizam-se no sistema com módulos operacionais e de governança. Como complemento, incluem-se em anexo dois relatórios gerados automaticamente pelo sistema: o relatório da camada Prata (dados tratados) e o relatório da camada Ouro (dados derivados por IA), documentando saídas representativas do processo. As figuras servem como evidência visual do funcionamento do *framework*, consolidando a conexão entre arquitetura, processamento e uso prático.

6 ANÁLISE DOS RESULTADOS OBTIDOS E COMPARAÇÕES COM OS TRABALHOS RELACIONADOS

6.1 Análise dos Resultados e Posicionamento frente aos Trabalhos Relacionados

Neste capítulo, avaliam-se os resultados do *framework* FIDLER com base nos critérios sintetizados a partir das dimensões discutidas no Capítulo 3. A subseção subsequente apresenta uma comparação qualitativa — sem pretensão de teste estatístico — com os trabalhos do estado da arte descritos no Capítulo 3.

6.1.1 Análise dos Resultados Obtidos

O *framework* FIDLER foi avaliado segundo critérios derivados das dimensões discutidas no Capítulo 3, que abrangem aspectos como integração de dados, governança, escalabilidade, suporte a análises temporais e uso de padrões abertos. A seguir, demonstra-se como o FIDLER atende a esses critérios:

1. **Integração de dados heterogêneos**: O FIDLER, baseado na arquitetura *Data Lakehouse*, integra dados em múltiplos formatos de alta frequência provenientes de fontes renováveis (eólica e maremotriz), oriundos de campanhas de medição ambiental, reconstrução de falhas e prospecções energéticas, conforme o modelo proposto por Armbrust [5]. Essa abordagem permite unificar armazenamento, metadados e análise em uma infraestrutura única, suportando múltiplos *workloads*;

2. **Governança e interoperabilidade**: Inspirado em Begoli [6] e Rain [8], o FIDLER adota os princípios FAIR (*Findable, Accessible, Interoperable, Reusable*) e governança federada com catálogos de metadados, garantindo rastreabilidade, transparência e prevenção de *data swamps*, com suporte a padrões abertos como os do *Open Geospatial Consortium* (OGC);

3. **Escalabilidade e organização**: Seguindo Janssen [7], o FIDLER utiliza camadas Bronze, Silver e Gold para organizar dados, promovendo clareza, escalabilidade e qualidade, com transformações automatizadas para análises eficientes;

4. **Suporte a análises temporais**: Baseado em Pohl [9], o FIDLER otimiza a gestão de séries temporais energéticas, suportando compactação, sincronização e geração de séries derivadas, embora apresente limitações em processamento em tempo real;

5. **Eficiência computacional**: Inspirado em Kumar [47] e Sun [11], o FIDLER oferece processamento distribuído e compactação avançada, garantindo eficiência em grandes volumes de dados, mas com desafios em cenários *offline*;

6. **Flexibilidade e adaptabilidade**: Conforme Schneider [24], o FIDLER suporta integração de fontes heterogêneas e ambientes distribuídos, com uma arquitetura adaptável, embora exija maior esforço técnico inicial;

7. **Inovação com inteligência artificial**: Diferentemente dos trabalhos relacionados, o FIDLER incorpora algoritmos de aprendizado de máquina para previsão, reconstrução de falhas e detecção de anomalias, voltados exclusivamente para dados de fontes renováveis, agregando valor às análises.

Os testes com dados reais das campanhas de Perizes, Parcel Manuel Luís e São João Batista (Maranhão), fornecidos pelo INEOF, indicam o potencial do FIDLER para suportar decisões estratégicas na transição energética e no planejamento de infraestrutura.

6.1.2 Comparação Qualitativa com os Trabalhos Relacionados

A Tabela 6.1 apresenta uma comparação qualitativa entre o FIDLER e os trabalhos relacionados, utilizando os critérios derivados do Capítulo 3. A coluna referente ao FIDLER é destacada em negrito. O objetivo é posicionar convergências e lacunas, sem pretensão de análise estatística.

Tabela 6.1 – Comparação qualitativa do FIDLER com os trabalhos relacionados.

Critérios	**[5]**	**[6]**	**[7]**	**[47]**	**[8]**	**[9]**	**[11]**	**[24]**	**FIDLER**
Integração de Dados	✓	✓		✓			✓	✓	✓
Governança FAIR		✓	✓		✓				✓
Escalabilidade			✓	✓			✓		✓
Análises Temporais						✓			✓
Padrões Abertos (OGC)	✓	✓			✓				✓
Compactação Eficiente						✓	✓		✓
Flexibilidade de Formatos								✓	✓
Uso de IA/ML									✓

Nota: ✓ indica presença ou suporte significativo ao critério, com base na análise qualitativa dos trabalhos.

Os trabalhos analisados — como Armbrust [5], Begoli [6], Janssen [7], Kumar [47], Rain [8], Pohl [9], Sun [11] e Schneider [24] — oferecem contribuições específicas, mas apresentam limitações, tais como complexidade de configuração, dependência de ferramentas ou ausência de suporte a análises temporais ou inteligência artificial.

O FIDLER destaca-se por integrar todas essas características em uma arquitetura unificada, com camadas de ingestão, armazenamento, processamento, metadados, análise e governança, além de incorporar inteligência artificial para previsão e reconstrução de dados, voltada exclusivamente para fontes renováveis. A análise aqui realizada é qualitativa e deve ser complementada por métricas empíricas (latência, custo, precisão) derivadas das execuções descritas no Capítulo 5, para consolidar o posicionamento definitivo.

6.1.3 Síntese

Este capítulo avaliou os resultados do FIDLER com base nos critérios derivados do Capítulo 3 e comparou-o qualitativamente com os trabalhos relacionados. O FIDLER atende a todos os critérios, incluindo integração de dados, governança, escalabilidade, análises temporais, uso de padrões abertos, compactação e flexibilidade, além de inovar com o emprego de inteligência artificial. Diferentemente dos trabalhos analisados, que frequentemente carecem de integração total ou suporte a inteligência artificial, o FIDLER unifica essas funcionalidades, oferecendo uma solução robusta e adaptável para a gestão de dados energéticos de fontes renováveis, com potencial significativo para aplicações na transição energética.

7 Conclusão

Esta dissertação propôs o FIDLER, um *framework* baseado na arquitetura *Data Lakehouse* para armazenamento e análise de dados de energia eólica e maremotriz. A seguir, apresentam-se os objetivos alcançados, as contribuições científicas e tecnológicas, as limitações identificadas, sugestões para trabalhos futuros e as publicações decorrentes.

7.1 Objetivos Alcançados

Desenvolveu-se a arquitetura do FIDLER, que gerencia grandes volumes de dados heterogêneos provenientes de campanhas de medição, reconstrução de falhas e prospecções energéticas. Essa solução possibilita análises técnicas e operacionais para auxiliar na tomada de decisões estratégicas e na pesquisa científica.

7.2 Contribuições Científicas e Tecnológicas

Esta pesquisa contribui cientificamente ao propor, formalizar e validar — por meio de um protótipo replicável — o FIDLER como um *framework lakehouse* orientado a dados de energias renováveis. O FIDLER integra hierarquia de domínio (Instituição–Campanha–Equipamento–Dataset), governança e auditoria, fluxo completo de ingestão/curadoria/publicação e suporte a inteligência artificial/aprendizado de máquina, preenchendo lacuna identificada na literatura quanto ao tratamento estruturado de séries temporais eólicas e maremotrizes em ambientes heterogêneos e operacionais.

As contribuições científicas e tecnológicas podem ser sintetizadas nos seguintes pontos:

- A arquitetura *Data Lakehouse* revela-se adequada para dados em múltiplos formatos e de alta frequência, facilitando análises temporais e espaciais;
- A estrutura em camadas Bronze, Silver e Gold garante rastreabilidade, qualidade e flexibilidade analítica com governança robusta de dados;
- O *framework* suporta *dashboards*, monitoramento e algoritmos de aprendizado de máquina para previsão, reconstrução de falhas e detecção de anomalias;
- Testes com dados reais das campanhas de Perizes, Parcel Manuel Luís e São João Batista (Maranhão), fornecidos pelo INEOF, indicam o potencial do FIDLER para apoiar decisões na transição energética e no planejamento de infraestrutura.

O FIDLER contribui para a tomada de decisões estratégicas em investimentos em parques eólicos e maremotrizes, podendo impactar direta e indiretamente uma matriz energética mais sustentável, apoiando a descarbonização do setor.

7.3 Limitações

O FIDLER requer intervenção manual em etapas como mapeamento inicial de atributos, definição de regras de negócio específicas e ajustes de parâmetros analíticos. As estruturas geradas preparam-se para análise, mas inferências específicas dependem de desenvolvimento adicional. Trata-se, portanto, de um *framework* semiautomático, adaptável a diferentes contextos regionais e tecnologias acessíveis. Equipamentos de fabricantes distintos demandam cuidado adicional para garantir a correta apresentação das análises.

7.4 Trabalhos Futuros

Com base nos resultados alcançados, o FIDLER apresenta elevado potencial para expansão e aprimoramento em diversas direções, visando ampliar seu impacto na gestão de dados energéticos e na transição para fontes renováveis. Propõem-se as seguintes sugestões para trabalhos futuros:

- Expandir o FIDLER com dados de outras regiões, adaptando-o a condições locais específicas;
- Desenvolver modelos avançados de aprendizado de máquina (LSTM, *Random Forests*, redes neurais) para previsão e reconstrução de dados;
- Criar modelos multicritério para avaliação de viabilidade técnica e financeira, utilizando métodos como AHP ou otimização;
- Integrar ontologias energéticas para maior interoperabilidade com padrões como OGC e FAIR;
- Implementar *pipelines* automatizados com ferramentas como Apache Airflow e Delta Lake;
- Construir *dashboards* interativos em Next.js ou Dash, direcionados a gestores e investidores;
- Gerar dados sintéticos para testar a resiliência do *framework* em cenários extremos.

7.5 Publicações

Os seguintes artigos forma publicados em decorrência dos trabalhos de pesquisa:

- Juan Ibanez, Rafael Veras, Henzo Costa, Arcilan Assireu, Denivaldo Lopes, Denisson Oliveira, Osvaldo Saavedra, Shigeaki Lima. Preliminary Wind Resource Assessment at the Mearim River Edge. In: XXV Congresso Brasileiro de Automática (CBA), 2024. Disponível em: https://easychair.org/publications/preprint/qVRw. Acesso em: 23 ago. 2025.
- Juan Ibanez, Isabelle Oliveira, Denivaldo Lopes, Shigeaki Lima, Osvaldo Saavedra. Desafios no Levantamento Seguro de Dados de Campo na Prospecção de Energias Renováveis. In: XI Simpósio Brasileiro de Sistemas Elétricos (SBSE), 2024. Disponível em: https://drive.google.com/file/d/1MVyE9VYqpzv4uLSLGpVfSOQ1AJ93TbHd/view. Acesso em: 23 ago. 2025.

Referências

[1] David Reinsel, John Gantz, and John Rydning. The digitization of the world: From edge to core. Technical report, International Data Corporation, 2018. Acessado em 26 de maio de 2025. Disponível em: https://www.seagate.com/files/www-content/our-story/trends/files/idc-seagate-dataage-whitepaper.pdf.

[2] International Energy Agency. Energy statistics data browser. Technical report, International Energy Agency, 2023. Acessado em 26 de maio de 2025. Disponível em: https://www.iea.org/data-and-statistics/data-tools/energy-statistics-data-browser.

[3] Empresa de Pesquisa Energética. Balanço energético nacional 2024: Ano base 2023. Technical report, Empresa de Pesquisa Energética, 2024. Acessado em 26 de maio de 2025. Disponível em: https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-748/topico-687/BEN2023.pdf.

[4] Stefan Partelow. What is a framework? understanding their purpose, value, development and use. *Journal of Environmental Studies and Sciences*, 13, 2023. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1007/s13412-023-00833-w.

[5] Michael Armbrust, Ali Ghodsi, Reynold Xin, et al. Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR)*, 2021. Acessado em 26 de maio de 2025. Disponível em: https://databricks.com/wp-content/uploads/2020/12/cidr_lakehouse.pdf.

[6] Edmon Begoli, Ian Goethert, and Kathryn Knight. A lakehouse architecture for the management and analysis of heterogeneous data for biomedical research and mega-biobanks. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 4643–4651, 2021. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/BigData52589.2021.9671534.

[7] Niels Janssen. The evolution of data storage architectures: Examining the value of the data lakehouse. Master's thesis, University of Twente, 2022. Dissertação de Mestrado (MSc Business Information Technology, Track: Data Science & Business), Faculty of Electrical Engineering, Mathematics & Computer Science (EEMCS). Acessado em 26 de maio de 2025. Disponível em: http://essay.utwente.nl/92801/1/Janssen_MA_EEMCS.pdf.

[8] Pablo Rain and Ania Cravero. Proposed metadata management model for data lakehouse. In *2024 43rd International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–8, 2024. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/SCCC63879.2024.10767658.

[9] Matthias Pohl, Nathira Dharindri Wijemanne, Daniel Staegemann, Christian Haertel, Christian Daase, Dirk Dreschel, Damanpreet Singh Walia, Arne Osterthun, Joshua Reibert, and Klaus Turowski. Data lakehouse for time series data: A systematic literature review. In *2024 IEEE International Conference on Big Data (BigData)*, pages 5833–5842, 2024. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/BigData62323.2024.10825961.

[10] Ahmed A. Harby and Farhana Zulkernine. Data lakehouse: A survey and experimental study. *Information Systems*, 127:102460, 2025. Acessado em 26 de maio de 2025.

[11] Bowen Sun, Xiaohan Yang, Ying Guo, Zhihao Zhao, Jing Chen, Hu Zhang, and Jibin Wang. Ocf-hp: An offline compaction framework based on hotspot prediction for data lakehouse. In *2024 9th International Conference on Electronic Technology and Information Science (ICETIS)*, pages 733–741, 2024. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/ICETIS61828.2024.10593781.

[12] Stefan Meisenbacher, Johannes Galenzowski, Kevin Förderer, Wolfgang Suess, Simon Waczowicz, Ralf Mikut, and Veit Hagenmeyer. Automation level taxonomy for time series forecasting services: Guideline for real-world smart grid applications. In Bo Nørregaard Jørgensen, Zheng Grace Ma, Fransisco Danang Wijaya, Roni Irnawan, and Sarjiya Sarjiya, editors, *Energy Informatics*, pages 277–297. Springer Nature Switzerland, Cham, 2025. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1007/978-3-031-74738-0_18.

[13] Juan Ibanez, Isabelle Oliveira, Denivaldo Lopes, Shigeaki Lima, and Osvaldo Saavedra. Desafios no levantamento seguro de dados de campo na prospecção de energias renováveis. In *Anais do XI Simpósio Brasileiro de Sistemas Elétricos (SBSE)*, 2025. Acessado em 23 de agosto de 2025. Disponível em: https://drive.google.com/file/d/1MVyE9VYqpzv4uLSLGpVfSOQ1AJ93TbHd/view.

[14] International Energy Agency. Data centres and data transmission networks, 2023. Acessado em 2 de julho de 2025. Disponível em: https://www.iea.org/energy-system/buildings/data-centres-and-data-transmission-networks.

[15] Emma Strubell, Ananya Ganesh, and Andrew McCallum. Energy and policy considerations for deep learning in nlp, 2019. Acessado em 26 de maio de 2025. Disponível em: https://arxiv.org/abs/1906.02243.

[16] David Patterson, Joseph Gonzalez, Urs Hölzle, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeffrey Dean. The carbon footprint of machine learning training will plateau, then shrink, 2022. Preprint. arXiv:2204.05149 [cs.LG, cs.AI, cs.GL]. Acessado em 26 de maio de 2025. Disponível em: https://arxiv.org/abs/2204.05149.

[17] Agência Nacional de Energia Elétrica (ANEEL). Caminhos da regulação: Histórico da legislação e da regulação do setor elétrico brasileiro, 2025. Acessado em 2 de julho de 2025. Disponível em: https://caminhosregulacao.aneel.gov.br/caminhos3.asp.

[18] J. D. F. Ibanez et al. Preliminary wind resource assessment at the mearim river edge, 2024. Acessado em 26 de maio de 2025. Disponível em: https://easychair.org/publications/preprint/15294.

[19] Ômega Energia S.A. Relato integrado 2022, 2022. Acessado em 2 de julho de 2025. Disponível em: https://srna.co/wp-content/uploads/2023/10/PT_Relato_Integrado.pdf.

[20] Empresa de Pesquisa Energética. Instruções para medições meteorológicas em parques eólicos: Nt epe-dee-re-057/2016, 2016. Acessado em 2 de julho de 2025. Disponível em: https://www.epe.gov.br/sites-pt/acesso-restrito/Documents/DEE_057_16_R4_Instru%C3%A7%C3%B5es_medi%C3%A7%C3%B5es_anemom%C3%A9tricas.pdf.

[21] Climate Policy Initiative. The challenge of institutional investment in renewable energy. Technical report, Climate Policy Initiative, 2013. Acessado em 22 de dezembro de 2025. Disponível em: https://www.climatepolicyinitiative.org/publication/the-challenge-of-institutional-investment-in-renewable-energy/.

[22] Microsoft. Microsoft azure cloud computing platform, 2025. Acessado em 22 de dezembro de 2025. Disponível em: https://azure.microsoft.com.

[23] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Boston, MA, 1994. Acessado em 26 de maio de 2025. Disponível em: https://www.pearson.com/en-us/subject-catalog/p/design-patterns-elements-of-reusable-object-oriented-software/P200000006370.

[24] Jan Schneider, Christoph Gröger, Arnold Lutsch, Holger Schwarz, and Bernhard Mitschang. Assessing the lakehouse: Analysis, requirements and definition. In *Proceedings of the 25th International Conference on Enterprise Information Systems (ICEIS)*, 2023. Acessado em 26 de maio de 2025. Disponível em: https://www.scitepress.org/Papers/2023/118405/118405.pdf.

[25] Kishore Reddy Gade. Data lakehouses: Combining the best of data lakes and data warehouses. *Journal of Computational Innovation*, 2(1), 2022. Acessado em 26 de maio de 2025. Disponível em: https://researchworkx.com/index.php/jci/article/view/3.

[26] Dipankar Mazumdar, Jason Hughes, and JB Onofre. The data lakehouse: Data warehousing and more, 2023. Acessado em 26 de maio de 2025. Disponível em: https://arxiv.org/abs/2310.08697.

[27] P. Magrino. A lakehouse-based platform for data-driven sustainability monitoring in energy-intensive production. In *CEUR Workshop Proceedings*, 2024. Acessado em 26 de maio de 2025. Disponível em: https://ceur-ws.org/Vol-3741/paper83.pdf.

[28] Maryann Xue, Yingyi Bu, Abhishek Somani, Wenchen Fan, Ziqi Liu, Steven Chen, Herman van Hovell, Bart Samwel, Mostafa Mokhtar, R. K. Korlapati, Andy Lam, Yunxiao Ma, Vuk Ercegovac, Jiexing Li, Alexander Behm, Yuanjian Li, Xiao Li, Sriram Krishnamurthy, Amit Shukla, Michalis Petropoulos, Sameer Paranjpye, Reynold Xin, and Matei Zaharia. Adaptive and robust query execution for lakehouses at scale. *Proceedings of the VLDB Endowment*, 17(12):3947–3959, 2024. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.14778/3685800.3685818.

[29] Ralph Kimball and Margy Ross. *The Data Warehouse Toolkit: The Definitive Guide to Dimensional Modeling*. Wiley, 3 edition, 2013. Acessado em 26 de maio de 2025. Disponível em: https://www.wiley.com/en-us/The+Data+Warehouse+Toolkit%3A+The+Definitive+Guide+to+Dimensional+Modeling%2C+3rd+Edition-p-9781118530801.

[30] Abraham Silberschatz, Henry F. Korth, and S. Sudarshan. *Database System Concepts*. McGraw-Hill, 7 edition, 2020. Acessado em 26 de maio de 2025. Disponível em: https://www.mheducation.com/highered/product/database-system-concepts-silberschatz-korth/M9780078022159.html.

[31] Fredrik Hellman. Study and comparison of data lakehouse systems. Master's thesis, University of Turku, 2023. Dissertação de Mestrado. Acessado em 26 de maio de 2025. Disponível em: https://www.doria.fi/bitstream/handle/10024/187408/hellman_fredrik.pdf?sequence=2.

[32] A. Czizeweski, F. M. Pimenta, and O. R. Saavedra. Numerical modeling of maranhão gulf tidal circulation and power density distribution. *Ocean Dynamics*, 70:667–682, 2020. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1007/s10236-020-01354-8.

[33] Jonas Vicente Pinto Junior, Clóvis Bôsco Mendonça Oliveira, Nadia Velez Parente, and Osvaldo Ronald Saavedra Mendez. Analysis of the potential of wind and ocean

energy in the state of maranhão. In *2018 13th IEEE International Conference on Industry Applications (INDUSCON)*, pages 509–516, 2018. Acessado em 26 de maio de 2025. Disponível em: https://ieeexplore.ieee.org/document/8627305.

[34] E. F. Codd. A relational model of data for large shared data banks. *Communications of the ACM*, 13(6):377–387, 1970. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1145/362384.362685.

[35] Martin Kleppmann. *Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems*. O'Reilly Media, Sebastopol, 2017. Acessado em 26 de maio de 2025. Disponível em: https://www.oreilly.com/library/view/designing-data-intensive-applications/9781491903063.

[36] W. H. Inmon. *Data Lake Architecture: Designing the Data Lake and Avoiding the Garbage Dump*. Technics Publications, 2017. Acessado em 26 de maio de 2025. Disponível em: https://technicspub.com/data-lake-architecture/.

[37] Microsoft. Azure blob storage: Object storage for the cloud, 2025. Acessado em 22 de dezembro de 2025. Disponível em: https://azure.microsoft.com/products/storage/blobs.

[38] United Nations. Transforming our world: The 2030 agenda for sustainable development, 2015. Acessado em 26 de maio de 2025. Disponível em: https://sdgs.un.org/sites/default/files/publications/21252030%20Agenda%20for%20Sustainable%20Development%20web.pdf.

[39] The Go Authors. The go programming language documentation, 2025. Acessado em 26 de maio de 2025. Disponível em: https://go.dev/doc/.

[40] Python Software Foundation. Python 3 documentation, 2025. Acessado em 26 de maio de 2025. Disponível em: https://docs.python.org/3/.

[41] Microsoft. Typescript handbook, 2025. Acessado em 26 de maio de 2025. Disponível em: https://www.typescriptlang.org/docs/handbook/intro.html.

[42] Vite Contributors. Vite documentation, 2025. Acessado em 26 de maio de 2025. Disponível em: https://vitejs.dev/guide/.

[43] Jinsheng Jiang, Haoran Ren, and Meng Zhang. A convolutional autoencoder method for simultaneous seismic data reconstruction and denoising. *IEEE Geoscience and Remote Sensing Letters*, 19:1–5, 2022. Acessado em 26 de maio de 2025. Disponível em: https://ieeexplore.ieee.org/document/9416991.

[44] Yifu Ren, Jinhai Liu, Jianan Zhang, Lin Jiang, and Yanhong Luo. A data reconstruction method based on adversarial conditional variational autoencoder.

In *2020 IEEE 9th Data Driven Control and Learning Systems Conference (DDCLS)*, 2020. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/DDCLS49620.2020.9275168.

[45] Z.-H. Hu and Y.-L. Song. Dimensionality reduction and reconstruction of data based on autoencoder network, 2009. Acessado em 26 de maio de 2025. Disponível em: https://www.jeit.ac.cn/.

[46] J. A. Restrepo-Carmona, J. C. Zuluaga, and M. Velásquez. Smart supervision of public expenditure: A review on data capture, storage, processing, and interoperability with a case study from colombia. *Information*, 2024. Acessado em 26 de maio de 2025. Disponível em: https://www.mdpi.com/2078-2489/15/10/616.

[47] Deeptaanshu Kumar and Suxi Li. Separating storage and compute with the databricks lakehouse platform. In *2022 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–2, 2022. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/DSAA54385.2022.10032386.

[48] Ahmed AbouZaid, Peter J. Barclay, Christos Chrysoulas, and Nikolaos Pitropakis. Building a modern data platform based on the data lakehouse architecture and cloud-native ecosystem. *Discover Applied Sciences*, 7:166, 2025. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1007/s42452-025-06545-w.

[49] Yu Meng, Qiang Li, and Ziheng Wang. Research on data lakehouse architecture for grid business data. In *Proceedings of the 2024 3rd International Conference on Energy, Power and Electrical Technology (ICEPET)*, pages 523–527, 2024. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/ICEPET61938.2024.10626461.

[50] Zhoujie Zhang, Lei Tao, Lin Xie, Shuzhou Wu, Ruili Ye, and Jiaqi Wang. High-performance fusion storage technology for massive power grid dispatching and operation data. In *2025 10th Asia Conference on Power and Electrical Engineering (ACPEE)*, pages 33–37, 2025. Acessado em 26 de maio de 2025. Disponível em: https://ieeexplore.ieee.org/document/11041192/.

[51] Fedor Nepsha, Vyacheslav Voronin, and Serge Kovalyov. Open data platform as a source of ml-driven innovations in the energy sector. In *Belarusian-Ural-Siberian Smart Energy Conference (BUSSEC)*, 2023. Acessado em 26 de maio de 2025. Disponível em: https://doi.org/10.1109/BUSSEC59406.2023.10296377.

Anexos

ANEXO A – Relatório automático da camada Prata (dados tratados)

Relatório LIDAR

gerado automaticamente

Informações da Campanha

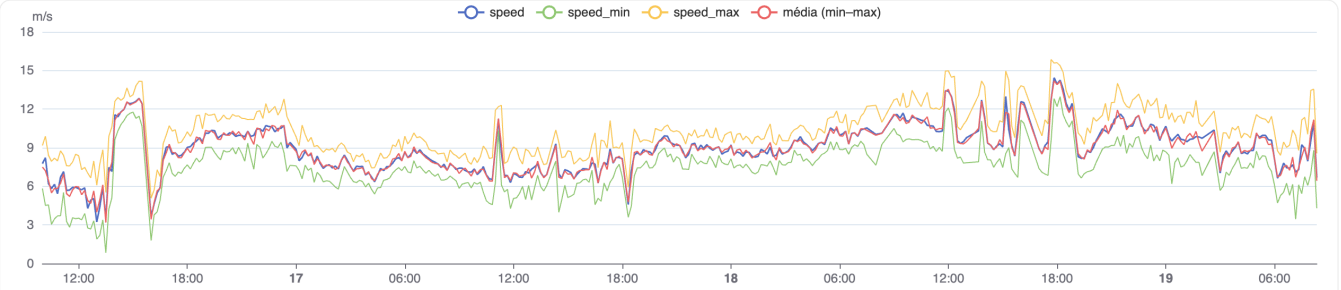
Campanha: Campanha Parcel Manuel Luís - Maranhão [1815b634-8462-4b63-9644-a378932cd170]
Código da campanha: CAMPPMLMA
Equipamento: LIDARZEPHI300M [e451d838-f3c9-44ed-84eb-bc541a633c37]
Janela de tempo: 16/06/2025, 10:00:00 → 19/06/2025, 08:30:00
Alturas (z) selecionadas: 40, 60, 80, 100, 120, 140, 160, 180, 200

Dados selecionados (preview – primeiras 10 linhas)

timestamp	freq	z	dir	speed
2025-06-16T13:00:00Z	00:10:00	40	103.063	7.054
2025-06-16T13:00:00Z	00:10:00	60	103.055	7.399
2025-06-16T13:00:00Z	00:10:00	80	102.437	7.625
2025-06-16T13:00:00Z	00:10:00	100	102.176	7.759
2025-06-16T13:00:00Z	00:10:00	120	102.99	7.843
2025-06-16T13:00:00Z	00:10:00	140	103.922	7.901
2025-06-16T13:00:00Z	00:10:00	160	104.78	7.972
2025-06-16T13:00:00Z	00:10:00	180	105.35	8.049
2025-06-16T13:00:00Z	00:10:00	200	105.946	8.121
2025-06-16T13:10:00Z	00:10:00	40	109.839	6.99

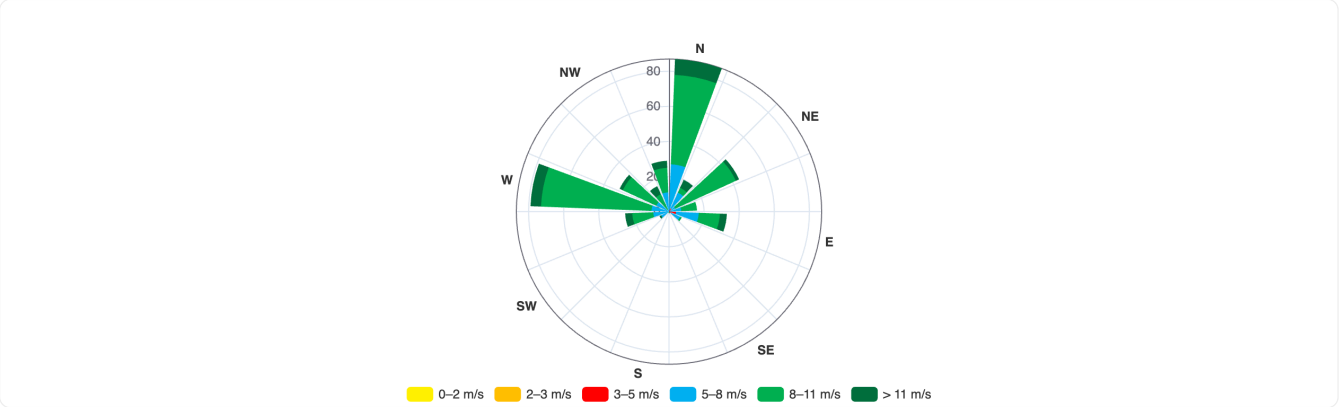
Gráficos

Velocidade × tempo



Direção × tempo

Rosa dos ventos



REWS — Rotor-Equivalent Wind Speed

O REWS é a velocidade do vento equivalente no rotor que, aplicada à curva de potência P = f(U), produz a mesma **potência média** que a integração sobre o disco do rotor.

Fórmula (conceito contínuo): REWS = $f^{-1}((1/A)\ \iint_A f(U(z))\ dA)$, onde f é a curva potência→velocidade e A é a área do rotor.
No relatório, isso é aproximado por fatias entre $z_{inferior}$ e $z_{superior}$ usando a lei de potência para o perfil vertical.

Principais usos: comparar condições medidas com a resposta energética; reduzir viéses da extrapolação simples; avaliar impacto do cisalhamento vertical na produção.

Parâmetros e resultados (gráfico)

- **z inferior:** 60
- **z superior:** 140
- **z vento:** 100
- **pontos REWS:** 397

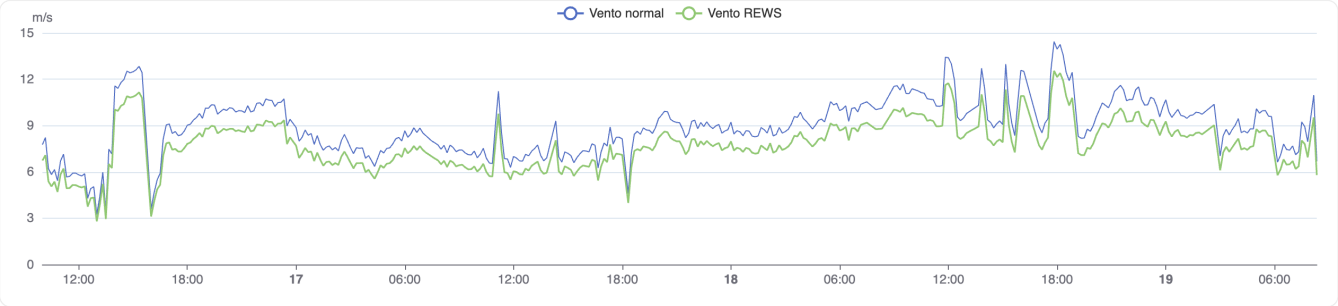


Tabela (amostra)

timestamp	z_bottom	z_top	z_wind	speed	speed_rews
2025-06-16T13:00:00Z	60	140	100	7.759	6.733744084092173
2025-06-16T13:10:00Z	60	140	100	8.218	7.074743086382136
2025-06-16T13:20:00Z	60	140	100	6.211	5.376792336706004
2025-06-16T13:30:00Z	60	140	100	5.826	5.060175259336774
2025-06-16T13:40:00Z	60	140	100	6.15	5.360959071469035
2025-06-16T13:50:00Z	60	140	100	5.424	4.698726897879761
2025-06-16T14:00:00Z	60	140	100	6.706	5.8254573871496875
2025-06-16T14:10:00Z	60	140	100	7.144	6.211045469104017
2025-06-16T14:20:00Z	60	140	100	5.67	4.937467838448716
2025-06-16T14:30:00Z	60	140	100	5.709	4.949054930666751
2025-06-16T14:40:00Z	60	140	100	5.912	5.145439443567868
2025-06-16T14:50:00Z	60	140	100	5.919	5.141051666026094
2025-06-16T15:00:00Z	60	140	100	5.812	5.049378638632851
2025-06-16T15:10:00Z	60	140	100	5.735	4.989308063641994
2025-06-16T15:20:00Z	60	140	100	5.875	5.060610899135682
2025-06-16T15:30:00Z	60	140	100	4.286	3.7507032960169537
2025-06-16T15:40:00Z	60	140	100	4.916	4.290581459444856
2025-06-16T15:50:00Z	60	140	100	5.041	4.336434813067469
2025-06-16T16:00:00Z	60	140	100	3.232	2.795235333077433
2025-06-16T16:10:00Z	60	140	100	4.391	3.847528657039758

Potência estimada

Gráfico (hub × REWS)

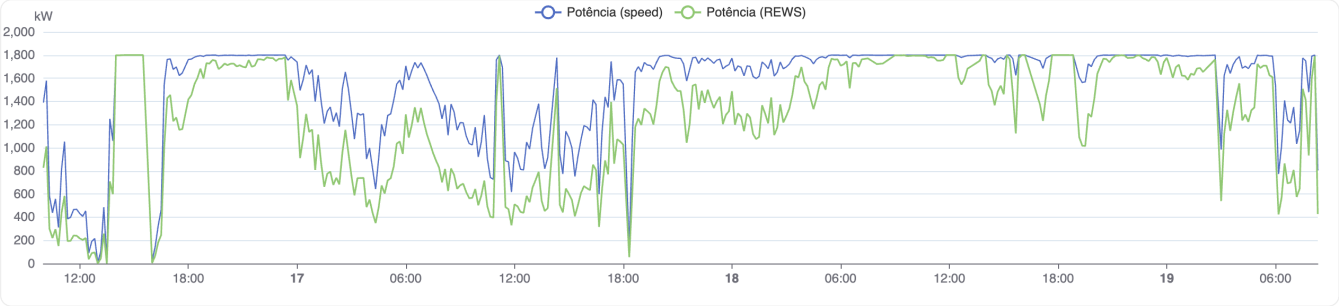


Tabela (amostra)

timestamp	z_bottom	z_top	z_wind	speed	rews	power_speed_kw	power_rews_kw	energy_speed_kwh	energy_rews_kwh
2025-06-16T13:00:00Z	60	140	100	7.759	6.733744084092173	1389.14	826.55	231.52	137.76
2025-06-16T13:10:00Z	60	140	100	8.218	7.074743086382136	1578.48	1010.59	263.08	168.43
2025-06-16T13:20:00Z	60	140	100	6.211	5.376792336706004	583.62	299.20	97.27	49.87
2025-06-16T13:30:00Z	60	140	100	5.826	5.060175259336774	437.58	220.04	72.93	36.67
2025-06-16T13:40:00Z	60	140	100	6.15	5.360959071469035	558.00	295.24	93.00	49.21
2025-06-16T13:50:00Z	60	140	100	5.424	4.698726897879761	311.00	150.77	51.83	25.13
2025-06-16T14:00:00Z	60	140	100	6.706	5.8254573871496875	812.12	437.40	135.35	72.90
2025-06-16T14:10:00Z	60	140	100	7.144	6.211045469104017	1052.84	583.64	175.47	97.27
2025-06-16T14:20:00Z	60	140	100	5.67	4.937467838448716	386.10	193.74	64.35	32.29
2025-06-16T14:30:00Z	60	140	100	5.709	4.949054930666751	398.97	195.83	66.49	32.64
2025-06-16T14:40:00Z	60	140	100	5.912	5.145439443567868	465.96	241.36	77.66	40.23
2025-06-16T14:50:00Z	60	140	100	5.919	5.141051666026094	468.27	240.26	78.04	40.04
2025-06-16T15:00:00Z	60	140	100	5.812	5.049378638632851	432.96	217.34	72.16	36.22
2025-06-16T15:10:00Z	60	140	100	5.735	4.989308063641994	407.55	203.08	67.93	33.85
2025-06-16T15:20:00Z	60	140	100	5.875	5.060610899135682	453.75	220.15	75.63	36.69
2025-06-16T15:30:00Z	60	140	100	4.286	3.7507032960169537	89.32	36.55	14.89	6.09
2025-06-16T15:40:00Z	60	140	100	4.916	4.290581459444856	189.88	89.87	31.65	14.98
2025-06-16T15:50:00Z	60	140	100	5.041	4.336434813067469	215.25	95.37	35.88	15.90
2025-06-16T16:00:00Z	60	140	100	3.232	2.795235333077433	8.35	0.00	1.39	0.00
2025-06-16T16:10:00Z	60	140	100	4.391	3.847528657039758	101.92	43.72	16.99	7.29

Estatísticas (última janela)

Janela	16/06/2025, 10:00:00 → 19/06/2025, 08:20:00
freq_s	600
speed_min	3.232
speed_max	14.433
speed_mean	8.912111
speed_p50	8.849
speed_p95	12.064
speed_std	1.805063
n_dir_ok	397
dir_mean_circ	352.91116
dir_std_circ	72.11066
resultant_r	0.452939
Atualizado em	21/12/2025, 17:36:02

Observações e limitações

- As imagens são “fotografias” dos gráficos no momento da geração do relatório.
- Se a cobertura válida estiver baixa, reavalie a janela, sensores e filtros de qualidade.
- Para uso comercial/energético, utilize a curva de potência oficial da turbina (ou SCADA do fabricante) e documente as condições de ar (densidade).

ANEXO B – Relatório automático da camada Ouro (dados derivados por IA)

Relatório — Camada Gold

Campanha: Campanha Parcel Manuel Luis
ID: 8c691b41-bf9b-46ef-b3d8-1ca9a73faece
Código da campanha: CAMPPMLMA
Modelo: cae1d-masked • Épocas: 180 • Gerado em: 2025-12-20T01:50:13Z

Resumo

Chave	Valor
anom_rate	0.06806282722513089
batch	64
epochs	180
hidden	8
loss_last	3.7592541909278836
lr	0.01
mae	0.5577821708443109
model	cae1d-masked
observed_count	382
reconstructed_count	26
rmse	0.8518278461463327
rows	408
window	24

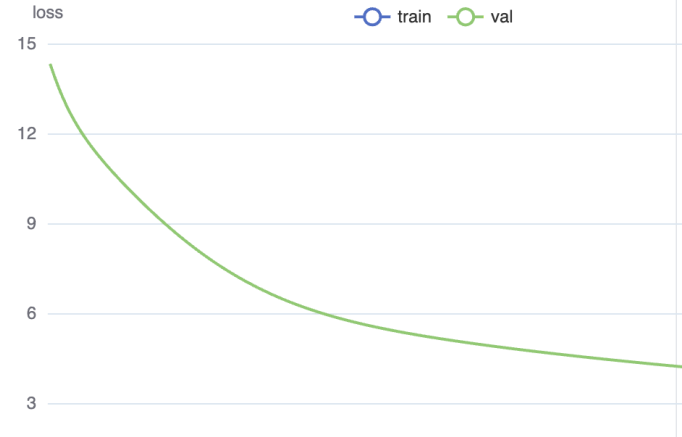
Análise automática

- **Reconstrução (MAE):** 0.558 m/s — **Boa** (0,5–1,0 m/s).
- **Reconstrução (RMSE):** 0.852 m/s — **Boa** (0,5–1,0 m/s).
- **Diagnóstico de treinamento:** Train e Val próximas e estáveis: sem sinais claros de overfitting.
- **Taxa de anomalias: 26/408 (6.8%). Dentro do esperado para dados estáveis.**
- Dicas: erros altos → ajuste `hidden_dim`/`epochs`, normalize melhor as features ou separe por regimes. Muitos falsos positivos → aumente `k` (MAD). Pouca detecção → diminua `k`.

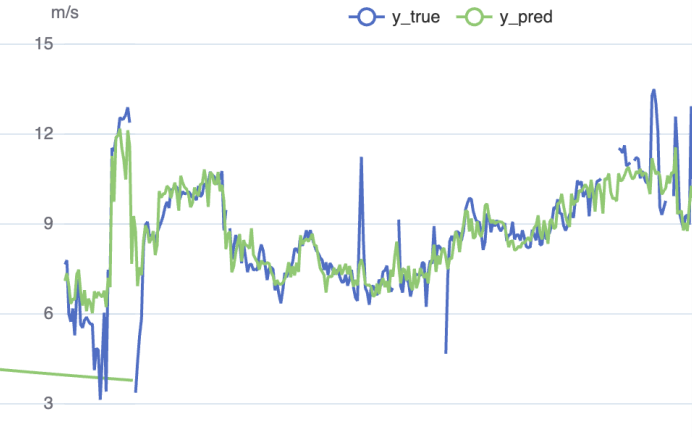
Explicação das métricas

- **Residual (y_true – y_pred):** diferença entre a velocidade real e a reconstruída.
- **MAE (Mean Absolute Error):** erro médio absoluto do residual. Quanto **menor**, melhor.
- **RMSE (Root Mean Squared Error):** penaliza mais erros grandes; também **quanto menor, melhor**.
- **MAD (Median Absolute Deviation) com k:** marcamos anomalia se `|residual − mediana| > k × MAD`.
  MAD é robusto a outliers; aumentar `k` gera *menos* anomalias; diminuir `k` gera *mais*.
- **Curva de loss (train/val):** indica estabilidade do treinamento (val próximo de train, sem crescer ao longo das épocas).
- **Reconstrução:** compara `y_true` e `y_pred` ao longo do tempo; aproximação boa indica padrão normal bem aprendido.

Curva de loss (train/val)



Reconstrução: y_true × y_pred



Métricas

Chave	Valor
anom_rate	0.06806282722513089
loss_last	3.7592541909278836
mae	0.5577821708443109
rmse	0.8518278461463327

Preview dos dados (15 primeiras linhas)

timestamp	z	speed	dir	w	ti	y_true	y_pred	residual	is_anomaly	quality
2025-06-16T13:00:00Z	80	7.625	102.437	-0.527	0.1358396	7.625	7.077306735145946	0.547693264854054	false	observed
2025-06-16T13:10:00Z	80	7.766	108.828	-0.476	0.1764219	7.766	7.334109848293805	0.43189015170619527	false	observed
2025-06-16T13:20:00Z	80	5.976	110.165	-0.131	0.1640514	5.976	6.9218909865571625	-0.9458909865571625	false	observed
2025-06-16T13:30:00Z	80	5.718	109.012	-0.259	0.1789494	5.718	6.315074640966551	-0.597074640966551	false	observed
2025-06-16T13:40:00Z	80	6.155	107.742	-0.45	0.150205	6.155	6.470730816441047	-0.3157308164410466	false	observed
2025-06-16T13:50:00Z	80	5.26	127.685	0.076	0.1618218	5.26	6.459456063483103	-1.1994560634831029	false	observed
2025-06-16T14:00:00Z	80	6.632	120.493	-0.543	0.1591609	6.632	7.280376237199853	-0.6483762371998534	false	observed
2025-06-16T14:10:00Z	80	7.133	125.05	-0.446	0.1074452	7.133	7.451172176310656	-0.3181721763106564	false	observed
2025-06-16T14:20:00Z	80	5.625	119.54	-0.231	0.1819824	5.625	6.720467133174595	-1.095467133174595	false	observed
2025-06-16T14:30:00Z	80	5.513	102.999	-0.19	0.1768399	5.513	6.253911682096461	-0.7409116820964607	false	observed
2025-06-16T14:40:00Z	80	5.767	105.166	-0.182	0.1844423	5.767	6.768436231028332	-1.0014362310283316	false	observed
2025-06-16T14:50:00Z	80	5.866	120.683	-0.579	0.1784524	5.866	6.065049879176399	-0.1990498791763997	false	observed
2025-06-16T15:00:00Z	80	5.734	101.722	-0.435	0.1738706	5.734	6.483124512353056	-0.7491245123530561	false	observed
2025-06-16T15:10:00Z	80	5.637	109.509	-0.266	0.1666739	5.637	6.324582984443159	-0.6875829844431598	false	observed
2025-06-16T15:20:00Z	80	5.634	125.665	-0.316	0.1785012	5.634	5.999046744733189	-0.36504674473318843	false	observed